Computer Aided Architectural Design Futures 2001

CAAD FUTURES 2001 PROCEEDINGS

Computer Aided Architectural Design Futures 2001

Proceedings of the Ninth International Conference held at the Eindhoven University of Technology, Eindhoven, The Netherlands, on July 8–11, 2001

Edited by

Bauke de Vries

Jos van Leeuwen

and

Henri Achten

Eindhoven University of Technology,
Department of Architecture,
Building and Planning,
Eindhoven, The Netherlands

SPRINGER-SCIENCE+BUSINESS MEDIA, B.V.

A C.I.P. Catalogue record for this book is available from the Library of Congress.

ISBN 978-94-010-3843-0 ISBN 978-94-010-0868-6 (eBook)
DOI 10.1007/978-94-010-0868-6

Cover designed by Jac de Kok,
Rendered images of the new faculty-building of Architecture designed by Bert Dirrix,
photographs of the conference venue at Eindhoven University of Technology
by Jos van Leeuwen

Printed on acid-free paper

All Rights Reserved
© 2001 Springer Science+Business Media Dordrecht
Originally published by Kluwer Academic Publishers in 2001
Softcover reprint of the hardcover 1st edition 2001
No part of the material protected by this copyright notice may be reproduced or
utilized in any form or by any means, electronic or mechanical,
including photocopying, recording or by any information storage and
retrieval system, without written permission from the copyright owner.

TABLE OF CONTENTS

Virtual Reality

CAAD Education

(Hyper) Media

Architectural Analysis

Urban Design

VR-DIS

PREFACE

Computer Aided Architectural Design has become a main factor in everyday processes in the architectural office. New developments in visualisation techniques, Internet, organisational forms, and programming languages push the boundary of what is technologically possible in CAAD software. Academic and industrial researchers alike are looking for new and unanticipated ways to use CAAD in a productive and creative manner. The CAAD Futures conference is a venue in which state-of-the-art work is presented and views for the future are discussed. Since its start in 1985, the bi-annual conference has been instrumental in charting the development of CAAD and use in architectural and engineering design. The series of conferences started in 1985 in Delft, and has since travelled through Eindhoven, Boston, Zurich, Pittsburgh, Singapore, Munich, and Atlanta.

The current proceedings provide the papers submitted for the CAAD Futures 2001 conference, which was held in Eindhoven. It counts over fifty papers, with contributions from many areas of research in Computer Aided Architectural Design. The breadth of the submissions is reflected by the sixteen categories that organise the proceedings. Most of these categories constitute well-defined areas of research, such as Case Based Reasoning, Virtual Reality, Hyper Media, Collaboration, and so forth. Others have proven more elusive: the category of Form Programming reintroduces the issues of description and generation of form by means of programming. Simulation appears in two categories: Simulation and Architectural Analysis. Papers in these categories differ mainly in the basic stance that is taken about the use of simulation results. XML is beginning to break through, but since it is so much integrated in the research work it did not stand out as a separate category.

The following paragraphs provide an overview of the papers per category.

Capturing Design

Capturing design is about the techniques and methods that support the designer in creating a digital representation of his/her mental image. For design recording, Gross, Do and Johnson propose to use speech recognition to create a searchable multimedia document, which includes the graphics as well as the designers' comments. Design creation and manipulation requires a more intuitive interface than the well-known WIMP interface. One approach is to interpret gestures [Kemp, Gross], another approach is to interpret digital sketches [Leclercq]. In the first case video is used to capture

hand positions and translate them into modelling commands, in the latter case pixels are converted into lines and relationships in the architects' sketch.

Information Modelling

Information modelling discusses methods for representing design information which captures architectural design features such as representational flexibility and temporal data. Sorts [Stouffs, Krishnamuri] specifies a common syntax, allowing for different vocabularies and languages to be created, and providing the means to develop translation facilities between these. Using such an information modelling method, Ekholm argues that user activities in a building should be taken into consideration by introducing an activity space with specific properties such as time schedule. With this information 3D models can be created which illustrate how these activities are accommodated in the building. Integration of spatial and temporal information is not only important in the design process but also in the construction process. Shih and Huang developed a system that supports presentation of changes in time as well as cross-analysis of spatial relationships between construction activities.

CBR Techniques

Case Based Reasoning has attracted many researchers as a method for architectural design knowledge encapsulation. On the urban level, Peng, Chang, Blundell Jones and Lawson implemented a system that can generate on user request, VRML models of city plans with its contextual information. On the building level, Chien and Shih developed a system that supports browsing the design space by filtering components and assemblies from the database. Heylighen and Neuckermans tested their web-based design assistant whether it really engaged students into in-depth explorations of designs from the case base.

Virtual Reality

VR technology has initiated a range of new developments in architectural design. The most wide spread application is the architectural walkthrough. Stappers, Saake and Adriaanse analysed CAVE experiments and conclude that narrative enhancements can substantially improve the level of engagement. VR requires special input devices for navigation. A very special device is the human muscle as used by Knight and Brown to activate signals. Maver, Harrison and Grant argue that the possibility to navigate

through virtual buildings is even more relevant for people with mobility or visual impairment. VR technology is taken one step further by Do into the direction of a design tool named VR sketchpad. This supports the recognition of simple shapes while sketching, translates them into building elements and displays the scene in VRML. An additional feature is the transparent window to quickly trace and extract any image from other application software.

CAAD Education

In CAAD education the Virtual Design Studio has received much attention. Clark, Maher and also Russell extend this idea into a Virtual Learning Environment. They argue that more important than form and space are the enhancements to give it a sense of place. Therefore special platforms are developed and used in architectural education. Architectonic learning and revealing new ideas is possible through digital narrative scenarios within a 3D space as presented by Strehlke and Engeli. Hyperstructures allow multi-threaded, multi faceted representations. Woodbury, Shanon and Radford introduce play to stimulate collaborative design. It builds confidence to continue and competence to perform. QaQish developed a method to evaluate the effectiveness and productivity of CAAD courses between and within students. Its purpose is to work as a framework to augment interactivity and positive learning.

(Hyper) Media

The medium through which architects work while designing, has undergone many changes and developments in the past years. The environment in which to communicate with others and oneself has become more responsive, intelligent, and geared to a natural interface. Cheng investigates the effectiveness and bias that several media induce when users are making surveys. This work gives indications how media such as photographs, video, sketching, etc. influence the recording process. Jung presents ongoing work on annotation of online geometric models by multiple participants to share design ideas. A fluent interface that is quick to use is required to make online-discussions easier to attend. Jung demonstrates how a pen-interface can work in such a system. Do shows a prototype of a system that can relate different design drawings by understanding the geometry of the drawings. It helps as an explorative tool to organise many drawings. Another way to look at drawings is presented by McCall, Vlahos, and Zabel. They show how the hand-made trace can provide a useful organising structure for hyperstructuring design drawings, in particular when based on

traces on drawings. In this way, the ease of hand sketching can be combined with automated organisation. Mishima and Szalapaj present a multi-media system for conveying architectural concepts to architects and architectural students. The content of the system focuses in particularly on analysis and design concepts.

Design Evaluation

Evaluation is a crucial step in the design process to assess the performance and impact of the building (component) design, and whether it lives up to the expectations of the designer, national codes, the client, etc. Automated code checking for example, has received some attention in the past. Yang and Li show how an Object Oriented approach for both representing codes and building designs can aid in automated code checking. The two other papers in this category deal with lighting evaluation. Moeck shows how a top-down approach, using evaluation criteria such as visibility, material appearance, and location of shadows and highlights can lead to a fast lighting proposition. This strategy reverses simulation techniques where first all sources need to be placed and then the lighting can be simulated and evaluated. Eissa, Mahdavi, Klatzky and Siegel test the hypothesis that lighting tools that use numeric simulation and scientific visualisation can provide adequate material for designer to evaluate the lighting design. Such research work is valuable to better understand and use visualisation tools in evaluation of designs.

Design Systems Development

The development of design systems is a complex task, in which the insights from information modelling, user interface, design methods and tools, etc. are incorporated and have to be balanced out. Two papers in this category deal with general issues in design systems development: usability and inconsistency. Chien demonstrates how an Object-Oriented Software Engineering strategy combined with a usability analysis method can help to develop prototype software with a higher degree of usability for the user. Hoffmann, Stumptner and Chalabi discuss the issue of consistency of data models. During the design process, they argue, multiple and also single designers cannot maintain consistent models of the design, as they are switching from perspective, design approach, organisation of the computer model, etc. The authors therefore propose a system that can hold these various views up to the point where they need to be unified again. In their paper, they also discuss the mapping functions between the various perspectives.

Collaboration

CAD support for multiple partners has been researched for quite some time now. Papers in this category focus either on the environment in which design partners collaborate [Johnson], [Han and Turner], or on the technical support issues that arise when working with multiple partners [Jeng], [Cerulli, Peng and Lawson].

Johnson points out that not only support for focused interaction is required, but also for unfocused interaction, which concerns the peripheral awareness of the designer of his or her partners. This information is important to convey in a design environment. Johnson presents a prototype interface that incorporates unfocussed interaction and present the first informal evaluations of such a system. Han and Turner experiment with VRML-based environments of distributed systems that support geometry display and communication with avatars of other designers, and show how these systems can be evaluated.

Jeng presents a rule-driven system that coordinates what kind of activities when, how, and by whom need to be undertaken in a design process with multiple participants. This coordination model is implemented as a web-based application. Jeng discusses the system as well as its potential use. Cerulli, Peng and Lawson present a system that records design processes, and that can provide meta-knowledge about this process for evaluation during or after the process. In this way, the decision-making process becomes more transparent for all partners involved, which can enhance the quality of decision making.

Generation

Automated building and shape generation have always been in the focus of research attention. In the past years, attention has been mainly on shape grammars. In this research field there are contributions from Park and Vakaló, and Chase and Liew. Park and Vakaló present a form-generating grammar that is derived during the design process by recording the objects that are involved in transformation steps. In this way, a designer does not to construct the shape transformation rules that are applied, yet the steps are recorded. Chase and Liew take a look at the field of redesign, where existing designs are adapted to suit changing requirements or tastes. They provide a framework based on feature grammars for this. Changes in style and appearance of the redesign are captured by means of rule modifications, using the mechanism of Function-Behaviour Structures.

Medjdoub, Richens and Barnard present a full-fledged automated generation system for pipe-routing of Low Temperature Hot Water plant

rooms. The system progresses through a sequence of steps based on a few starting assumptions for the room and provides a layout for the piping. The result can be manipulated interactive, while the system tries to conform to the constraints that apply to the piping system.

Design Representation

Representation of design plays a very central role in the discipline of architecture. Not only is the representation of what is (being) designed an inherent part of the design and construction process, also in architectural history and theory, design representation has a key-role. Augmenting the richness of design representations is the objective of Tunçer, Stouffs, and Sariyildiz. This objective is pursued by interpretation of design documents, decomposition of their content, and integration of decomposed documents into a rich information structure. Two different implementation approaches are discussed that are used to represent this type of rich information structures.

Kulinski and Gero regard design representation as a triplet of function, behaviour, and structure. The transitions between these three states of a design can be represented in a graph. The authors concentrate on analogy as a strategy in the design process; the matching of domains that share common representational structures. A model of situated analogy for design is presented, involving the re-representation of the knowledge in a domain for the purpose of finding analogies.

Tang and Gero argue that current CAAD systems lack the ability to support the speed and vagueness of the conceptual designing process. Through empirical data they show that perceptual cognition and connections between sketches and higher level cognitive functions are as important as the explicit representations that are recorded by CAAD systems. This leads to the conclusion that design representation should not be fixed but allow for the emergence of a variety of unexpected forms and spatial relationships.

Knowledge Management

Closely related to the subject of Design Representation is the topic Knowledge Management. This area, in the context of design support, addresses the development of methods for storing, retrieving, interpreting, and structuring design knowledge. Ciftcioglu and Durmesevic describe an approach of information mining to handle the complex relationships among information components and the interdependencies in information structures.

Turk, Cerovsek, and Martens attempt to use machine learning and data mining techniques to cluster topics in the area of CAAD. This attempt has

the objective to provide an intelligent search interface to the CUMINCAD online database of CAAD publications. A number of algorithms for automatic analysis and grouping of publications are investigated, albeit with insufficient success.

Mann and Ó Catháin present how TRIZ, a systematic method for support of innovation and creativity, can be applied in the discipline of architectural design. Through a number of case studies, they demonstrate the potential of this method for re-using design knowledge in and outside the discipline of its original context.

Form Programming

Form programming comprises of a group of papers that focus on the generation of form by means of some algorithmic approach, other than shape generation. Gross presents a low level programming language with which one can quickly generate forms. As the author claims, advanced and complex shape modelling in CAD systems still is cumbersome, and some shapes cannot be made at all. A different approach to form programming is demonstrated by Choi, Kwon, and Lee. They have developed a design buffer which stores the rapid production of design ideas in the early design phase, and only later requires the architect to focus on a more rigid computer model, drawing from the many ideas and shapes in the design buffer. A Boolean set operator functions on the objects in this buffer, allowing the architect to quickly interact with the ideas. Mahalingam has developed a system in which effectors work on geometry (or other effectors) taking into account particular goals or constraints. By having effectors in particular for interior and exterior conditions of an envelope, shapes for example of an auditorium can be derived from this balance of forces. The work presented by Gavin, Keuppers, Mottram, and Penn, has a twofold target: one, to present a virtual working environment for multiple participants, and two, a mechanism to generate the environment in such a way that it reflects the design history and current status of the users in the environment. For this last purpose, the authors have developed a generative algorithm for the environment which gets feedback from analysis and user responses and behaviour.

Simulation

In the simulation section, four performance aspects are highlighted, namely: structural analysis, airflow analysis, HVAC analysis and life cycle analysis. Martini proposes a particle system approach for real-time, non-linear physical simulation. Koutamanis and den Hartog introduce surfaces to

define the interaction between the simulation of indoor climate and the representation of its spatial form. Mahdahvi, Brahma and Gupta tackle the problem of early design simulation by using a differential building representation in combination with agents that provide reference designs for technical subsystems. Ries and Mahdahvi developed an affordance impact assessment method for regional environmental simulation. All researches have in common that building performance simulation is integrated in a design evaluation tool.

Architectural Analysis

In contrast with the previous section, there are no computational models for architectural analysis that can be derived from the laws of nature. Yet there is a strong interest in design support on the architectural level. Analysis after the design process can bring forward new design knowledge. Koutamanis, van Leusen and Mitossi obtained new insights on activity allocation and compartmentalization by analysing pedestrian circulation. A very special method to enrich architectural investigation is introducing faking as part of scenario-building [Brown], currently well supported by digital techniques. Analysis as part of the design process may benefit from the curious agents proposed by Saunders and Gero that will support you in discovering interesting designs. Affordances of designs from the occupants' perspective is yet another typical architectural design consideration that is used by Tweed to analyse the use-value of a design for a variety of different occupants.

Urban Design

Urban plans can be reconstructed from the existing situation or newly designed using a design support tool. Chevier and Perrin developed an interactive tool for virtual 3D rough reconstruction of buildings using a scanned photograph and a cadastral plan. Having such a 3D model, people can participate in landscape design with a web-based tool developed by Alpha and Iki and Zhang, Tsou and Hall, thus creating design alternatives and evaluating the visual experience. Designing new urban plans is supported by the tools of Chitchian, Sauren, Sariyildiz and Heeling and of Canaparo and Robiglio. In the first case the tool is customized with specific objects and design scales. In the second case the suburbanization process is simulated using cellular automata and genetic programming.

To conclude, there are no clear revolutions going on in the field, although hindsight might prove us wrong in this respect. What we have here is an overview of a mature and well-established line of inquiry which boosts a rich development of tools to support architectural design with computers.

The CAAD Futures 2001 Organising Committee
Bauke de Vries, Jos van Leeuwen, Henri Achten

The CAAD Futures Secretariat
Marlyn Aretz

The Design Amanuensis

An Instrument for Multimodal Design Capture and Playback

Mark D. Gross, Ellen Yi-Luen Do, and Brian R. Johnson
Design Machine Group, University of Washington

Key words: Protocol analysis, recording, design process research

Abstract: The Design Amanuensis* supports design protocol analysis by capturing designers' spoken and drawing actions and converting speech to text to construct a machine-readable multimedia document that can be replayed and searched for words spoken during the design session or for graphical configurations.

1. INTRODUCTION

Design researchers have used think-aloud, or protocol analysis, studies of designers in action to better understand their design processes. Protocol analysis studies are used across the domains in which cognitive scientists seek to understand human problem-solving. Protocol analysis (Newell, 1968) was brought to the attention of the emerging fields of cognitive science and artificial intelligence in 1972 by Herbert Simon and Allen Newell's important book, "Human Problem Solving" (Newell, 1972). Newell and Simon were among Carnegie Mellon's most prominent scientists and this method of protocol analysis was employed in some of the first empirical studies in design research by Carnegie Mellon researchers Eastman (Eastman, 1968) and later Akin (Akin, 1978). Donald Schön based his analysis of design as an example of reflection-in-action on an analysis of protocols taken in architecture design studios at MIT (Schön, 1985). Many

* amanu**ens**is: Latin, from *a manu* slave with secretarial duties; one employed to write from dictation or to copy manuscript (Merriam Webster Collegiate Dictionary)

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 1–13.
© 2001 *Kluwer Academic Publishers.*

subsequent research studies (e.g., Goel's Sketches of Thought (Goel, 1995); Goldschmidt's studies of visual reasoning in design (Goldschmidt, 1992)) have employed the study of think-aloud protocols, including a recent workshop (Cross, Christiaans and Dorst, 1996) that brought together a diverse collection of research teams who studied the same design protocol to compare their analysis and conclusions. Think-aloud protocol analysis has some reported shortcomings, notably that talking aloud interferes with performing the task at hand (Wilson, 1994); nonetheless it is widely accepted as a useful method of design research.

In a typical think-aloud study designers are given a problem and then they are observed, audio-taped, and often videotaped as they work the problem. Later, the design researcher transcribes the taped record of the design session and using his or her notes constructs a time labelled record of the events in the session. The study may examine a single designer working alone or a team of designers working together. The record typically consists of the drawings made and the designers' comments; in addition it may indicate gestures such as pointing to portions of the drawing or body movement. This transcript is typically the main corpus of research material used for further analysis.

Many researchers proceed from the transcript to make time-based representations of the design history, from which they reason further about the design process. These 'design scores' take various forms, for example Goldschmidt's Link-O-Gram diagrams (Goldschmidt, 1990). These more abstract representations of the design process may be more directly useful in analysing design behaviour, but they also are abstractions or translations of the raw protocol data captured during the design session.

Valuable as think-aloud studies are in design research, they also require a great deal of tedious effort on the part of the researcher. One of the more laborious tasks in conducting a protocol study is transcribing the design session after it has been recorded on tape. Potentially relevant events in a design session often happen every few seconds, and co-ordinating the times of events in different modalities is important. For example, based on a protocol analysis study Akin and Lin report that novel design decisions usually occurred when the designer was in a "triple mode period": drawing, thinking, and examining (Akin and Lin, 1995). This type observation can only be made based using a time-accurate record of the design session.

Depending on the level of detail that the researcher is interested in, a one-hour design session can demand tens of hours of transcription time. The researcher must carefully watch a video tape frame-by-frame to locate start and end times for events and transcribe the words spoken during the session. A single design session may result in a transcript that is tens of pages long.

We (Gross and Do) first realised the need for a design recording instrument when Do was capturing data for her doctoral dissertation (Do, 1998), which involved transcribing and analysing videotapes of four one-hour design sessions. A great deal of effort was spent locating the precise times of spoken comments, drawing marks, and associating them with drawings made on paper, or images of drawings recorded on video tape. Although several research efforts have tried to support recording and analysis of multimodal conversations, no off-the-shelf tools were readily available for our purpose.

The Design Amanuensis project aims to facilitate this transcription, by helping capture the spoken and drawn events of a design session, and by constructing a machine-searchable transcription that serves as a pointer into the source data captured during the original design session. We have built a first working version of this system that captures the designer's drawings using a digitising tablet and pen as well as the spoken think-aloud protocol. The captured audio is run through an off-the-shelf speech recogniser to generate a text transcription of the design session. The three components: drawing, audio, and text transcript, are arrayed in an interactive multimedia document. The design researcher can then review, correct, and annotate this document to construct a transcript of the design session.

As current speech-recognition software is somewhat unreliable, it is important to allow the design researcher to repair the machine-made text transcript; nevertheless, starting with an initial transcription is a significant improvement over starting from scratch. The speech recognition software associates the text transcript with the corresponding recorded audio so these repairs are easier to make than working with an ordinary audio or video recording.

The paper reports on the design and implementation of the Design Amanuensis. We review some of the specific challenges in constructing this system and how we addressed them, as well as our plans for enhancing this system in the future. Finally, we discuss other applications for the Amanuensis, including its application as a note-taker for distributed design meetings that take place over the Internet.

2. RELATED RESEARCH

Many efforts to build recording and indexing systems for mutli-modal information (e.g (He, Sanocki, Gupta et al., 1999)) are designed for automatically indexing streamed video, for example newscasts and lectures. Others are designed to "salvage" meetings; to create a useful, searchable record of a multimodal conversation taking place at a whiteboard or around a

table. Still other "personal notebook" efforts focus on studying or supporting the individual note-taker engaged in a specific task, for example, listening to a lecture.

In light of the relatively small number, world wide, of design researchers, it is unsurprising that we could find only one project specifically designed to record and index design protocols. With much the same goal as we have in mind, McFadzean, Cross, and Johnson (McFadzean, 1999; McFadzean, Cross and Johnson, 1999), have developed an experimental apparatus called the Computational Sketch Analyser (CSA), which they use for capturing design protocols. The CSA records drawing marks on a digitising tablet, classifies, time stamps, and co-ordinates them with a videotaped record of the drawing activity. However, the CSA does not transcribe the think-aloud protocol, leaving this task to the researcher, nor does it enable the researcher to search the record. In reviewing a design protocol that may extend to fifty or more transcribed pages, it is useful to be able to search the record for specific words or phrases, as well as specific drawing marks.

Oviatt and Cohen's Quickset program aims to support multimodal conversations, for example, military mission planning (Oviatt and Cohen, 2000). Quickset's important contribution to Human-Computer Interface research is using context from one mode (e.g., speech) to aid recognition in another mode (drawing). However, the Quickset system does not record speech continuously and independently of drawing acts. Rather, each speech event is associated directly with a drawing event: audio recording for that event begins when the pen goes down. Continuous and independent speech and drawing are typical in design protocols, and so Quickset would not serve our purpose.

Stifelman's Audio Notebook (Stifelman, 1996) was designed as a note-taking aid for a student attending a lecture. The student's handwritten notes were recorded using a digitising tablet and the lecture was simultaneously recorded in an audio file. The audio track and hand written notes were co-ordinated using the time stamps, so that a student could review the lecture by pointing to the notes he or she had taken, which served as an index into the audio track. The Audio Notebook did not perform speech recognition, so the record was indexed by marks made on the pad, but it could not be searched.

The Classroom 2000 project (Abowd, Atkeson, Feinstein et al., 1996; Abowd, 1999) used ubiquitous computing concepts to capture lectures, instructor's whiteboard notes, and student notes on personal handheld devices, and to later enable students to access these records of a class to enhance learning. Initially focused on capture and replay of lectures and notes, the project has recently explored using commercial speech recognition software to transcribe spoken lecture material.

Moran's Tivoli project has explored capture and indexing of spoken audio and whiteboard drawing to support 'salvaging'—an activity involving "replaying, extracting, organizing, and writing"—the records of meetings. (Moran, Palen, Harrison et al., 1997). This project also did not attempt to transcribe the spoken audio, but provided a case study of how the system was used over an extended period of time.

3. DESIGN PROTOCOLS

3.1 Example of a design protocol

Figure 1 shows a typical excerpt from a design protocol taken without the aid of the Design Amanuensis (Do, 1998). Entries in the leftmost column contain frames clipped from a videotape made during the session, in which the content was aimed at the designer's work area. The second column indicates the elapsed time in minutes and seconds since the session began. The third column records the designer's spoken comments (e.g., "ummm... what if we put the main entry here..." as well as the designer's physical actions (e.g., "picks up pencil and begins to draw rectangles"). The fourth column contains the researcher's annotations, and the rightmost column contains scanned excerpts of the design drawing made at this point in the protocol.

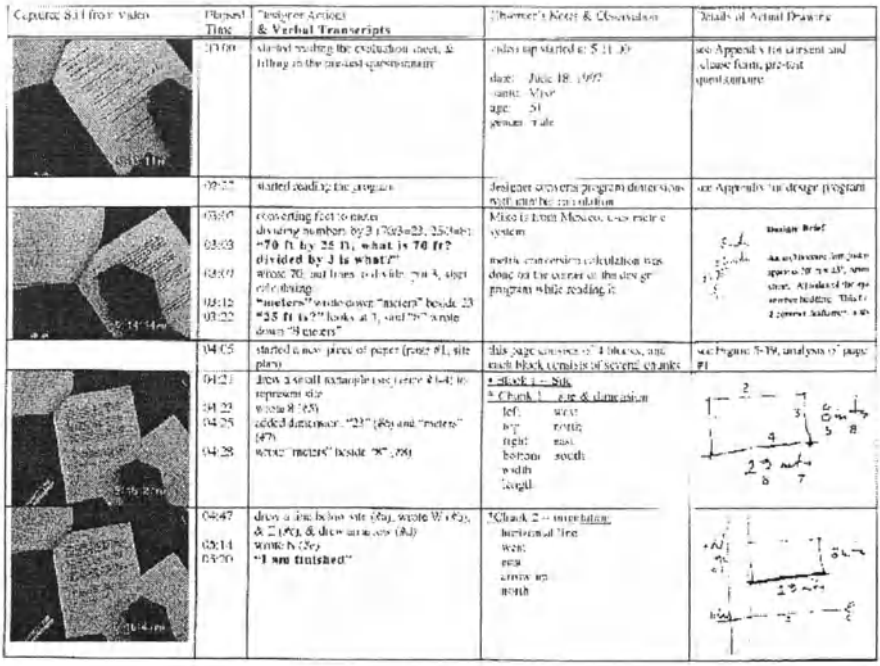


Figure 1. Excerpt from a design transcript showing frames from videotape, transcribed think-aloud protocol, and design actions

3.2 Recording a design protocol

The Design Amanuensis helps create this type document by recording all drawing and audio data as it is produced. The designer draws on an LCD digitising tablet and all spoken remarks are recorded as digital audio. Although the digitising pen and tablet are quite comfortable and natural to use, some time is required for the designer to become familiar with this mode of drawing. Similarly, the speech-to-text software works best when trained for the individual speaker. Once the designer is familiar with the drawing environment and the speech recognition software has been trained, the design recording begins. As the designer draws and talks, the program records these actions into files.

3.3 Reviewing the design history

When the design session is finished, the Design Amanuensis processes the files and makes them ready for replay. Then, the researcher can play back the entire design session from the beginning, or any portions of it. The

researcher can move forward and backward through the session (using a tape-recorder style control panel), or search for words in the text transcript or for drawn figures. The researcher can also point to a specific element of the drawing to review the corresponding portion of the design record. As the researcher reviews the record, he or she can correct erroneous text transcriptions by simply selecting and retyping them as in a text editor, as well as add annotations to the record.

Figure 2 shows the simple player interface for browsing the design record. Interface buttons enable "rewinding" to the beginning of the session, moving backward or forward one (speech or drawing) event at a time, selecting an element from the drawing to set the playback session clock as well as loading new session data and establishing a time mark. The three numbers display the start timestamps of the most recent previous event (left), the current event (centre) and the next event (right).

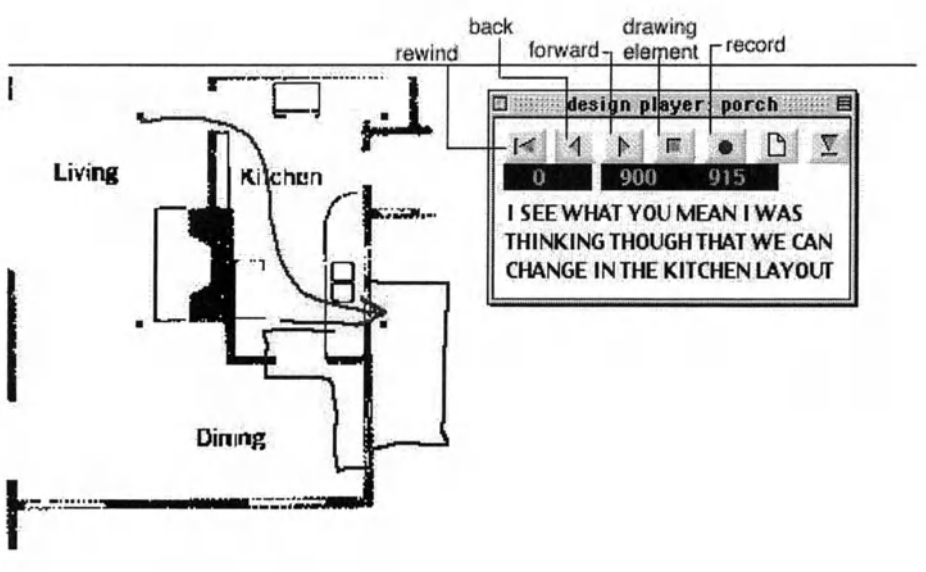


Figure 2. Playing back part of a design conversation

4. IMPLEMENTATION

The Design Amanuensis is implemented on Macintosh computers using a combination of software components. Drawing capture and management is handled by an extension of an existing research software program, the Electronic Cocktail Napkin (Gross and Do, 2000). Speech capture and recognition is handled by an off the shelf commercial application, IBM's

ViaVoice™ for Macintosh. Audio file format translation is handled by Norman Franke's excellent freeeware utility, SoundApp PPC and inter-application control is handled by Apple's AppleScript language. All these components are co-ordinated by code written especially for this purpose in Macintosh Common Lisp.

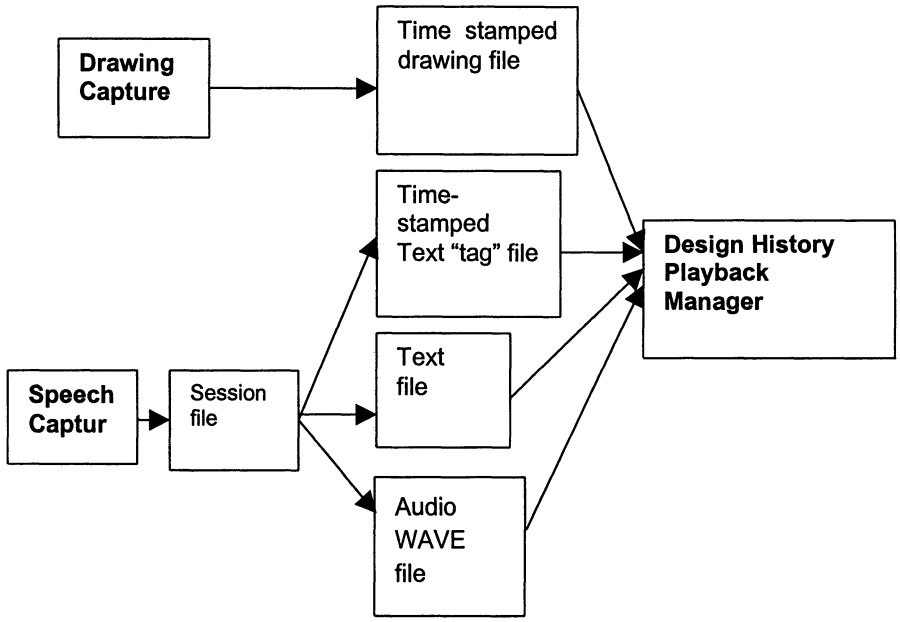


Figure 3. System diagram of the Design Amanuensis showing the two capture modules (drawing and speech capture), the four main files these modules create, and the design history playback module.

4.1 Drawing capture

The drawing capture component of the Amanuensis is an extension of the Electronic Cocktail Napkin. This program records each drawing mark as it is made, and records with each mark the time when it was initiated. In addition to recording the (x, y) co-ordinate stroke information for each drawing mark, the Electronic Cocktail Napkin attempts to recognise the marks as they are drawn. The Napkin program produces a drawing file that contains the entire set of time stamped drawing marks.

4.2 Speech capture and transcription

The Amanuensis uses IBM's commercial product, ViaVoice™ for Macintosh, for speech capture and transcription. This program records audio and transcribes the audio record to text. ViaVoice produces a single "session file" that contains the recorded audio, the text transcription, and tags that indicate the starting and ending times for each identified word and phrase in the audio, as well as alternative transcriptions of each word. The Amanuensis parses this single large file into several smaller component files (Figure 3). It also uses a helper application (SoundApp PPC) to convert the audio from the WAVE format that ViaVoice™ produces to the Mac native AIFF format.

4.3 Initiating a design session recording

Initiating a design session recording is straightforward: The program must simultaneously start recording audio and zero the clock for the Cocktail Napkin drawing program. In our current version these applications run in parallel on separate machines, because the speech recording software requires a dedicated processor. An AppleScript™ program launched from the Common Lisp environment tells ViaVoice™ to start recording and the Napkin program zeroes its clock. When the session is over, a similar AppleScript is launched to tell ViaVoice™ to stop recording and to save the session in a file. Once the file is saved, the Amaneuensis co-ordinating software parses ViaVoice™'s session file into its several components, again using AppleScript to call on SoundApp PC to convert the WAVE format audio to the Mac native AIFF. Once this is accomplished, the design session is ready for playback.

4.4 Design history playback

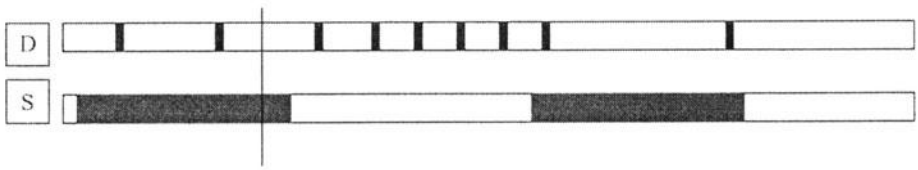


Figure 4. A diagram of the drawing (D) and speech (S) tracks of a design history. Several relationships are possible between the start and end of a speech event and the previous and next drawing events.

Figure 4 illustrates the two tracks of the design history. The drawing track consists of discrete events that have a single time-stamp, the time of the pen-down event that initiated that drawing mark (indicated by vertical lines in the "D" track). The speech ("S") track consists of continuous segments of speech alternating with silence (indicated by blocks in the track). A thin vertical line crossing the two tracks indicates the 'current time' being replayed on the recorder.

Events on the two tracks may have several different relationships: The designer may have been drawing silently, speaking without drawing, or drawing while speaking. Replaying the record from the beginning of the session (or any given time) is straightforward: set the clock to the desired time, find the nearest events in the speech and drawing record, and begin replaying the two tracks in time sequence. To ensure that the speech track remains intelligible, the speech track is always replayed starting at the beginning of the word that was being spoken.

For careful study of the record it is useful to 'single step' through the design history, moving forward one event at a time. When the designer presses the forward button on the playback console, the Design Amanuensis examines the audio and drawing tracks to determine which contains the next event. It then plays that event, displaying the text and playing the audio if the next event is on the speech track, or revealing the next drawing mark if the event is on the drawing track.

Specifically, if the next event is a drawing event, the Amanuensis will play (or replay) any speech event whose start and end times bracket the drawing event time. If the next event is a speech event, the Amanuensis will play any and all drawing events that occurred between the start and end times of the speech event.

It is also useful to be able to point at a mark on the drawing, and review the conversation that was taking place at that time (a common feature in audio indexed notebook systems). When the reviewer selects a drawing mark and requests to hear the audio, the system sets the clock to the timestamp associated with the drawing mark and plays a single event at that time.

Finally, the system supports both text and graphical search. The text search feature enables the reviewer to search for a word or phrase in the transcribed record, and set the playback clock to the time associated with that phrase. For example, the reviewer could request to review the design record associated with the phrase "light from the west." The system would find the point in the design session where these words were spoken and set the playback clock to the appropriate time. Similarly, the reviewer can use a graphical search to find parts of the record where a particular drawing was made. The Electronic Cocktail Napkin's diagram similarity matching is

employed to find parts of the drawing that are similar to the search sample the reviewer provides, and then the playback clock is set to the time stamp associated with the part of the diagram found.

5. FUTURE WORK

Our main goal is to use the Design Amanuensis as a tool for design protocol analysis in empirical studies of designers in action. We plan to develop and refine the capabilities of the program for this purpose, as we have found design protocol studies useful in understanding what designers are doing, but we have been inhibited in conducting such studies by the sheer amount of data to be managed and the lack of appropriate tools to manage this data.

We do, however, see a wider potential application of this work. An application of the Design Amanuensis is to record design conversations, both where the designers are co-located and where several designers who are geographically distributed collaborate via the Internet, drawing, writing, and talking. Our research laboratory space was recently renovated, and over the course of six months we participated in a number of design meetings at which drawings were reviewed, changes suggested, and decisions taken. On several occasions we wished to review the meetings (some of which lasted two hours), and specifically to recall what had been said about a particular topic or physical decision. A searchable digital record would have been useful.

An obvious addition to the current system is co-ordinating a video record of the design session. This would enable design researchers to include in the record gestures made over the drawing—which often include references to drawn elements—as well as body language that the current system does not capture. Time-stamped video would also be valuable for the distributed meeting support application of the system.

Finally, going beyond the immediate applications of protocol analysis and distributed meeting support, the system suggests an approach to constructing informative—yet informally structured—hyperdocuments that include mark-up and annotation on traditional design drawings, that record spoken and transcribed design rationale, in a way that can be browsed by people and searched by machine.

6. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant No. IIS-96-19856 and IIS-00-96138. The views contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

7. REFERENCES

Abowd, G. D., 1999, "Classroom 2000:An experiment with the instrumentation of a living educational environment." *IBM Systems Journal* 38(4), p. 508-530.

Abowd, G. D., C. G. Atkeson, A. Feinstein, et al., 1996, "Teaching and Learning as Multimedia Authoring: The Classroom 2000 Project", In *ACM Multimedia '96*, ACM, Boston, p. 187-198.

Akin, O., 1978, "How Do Architects Design", In E. J.-C. Latombe (Eds.), *Artificial Intelligence and Pattern Recognition in Computer Aided Design, IFIP*, North-Holland Publishing, New York, p. 65-104.

Akin, O. and C. Lin, 1995, "Design Protocol data and novel design decisions." *Design Studies* 16(2), p. 211-236.

Cross, N., H. Christiaans and K. Dorst, eds., 1996, *Analyzing Design Activity*, John Wiley & Sons, New York.

Do, E. Y.-L., 1998, The Right Tool at the Right Time: Investigation of Freehand Drawing as an Interface to Knowledge Based Design Tools, Georgia Institute of Technology, .

Eastman, C. M., 1968, "On the Analysis of Intuitive Design", In G. T. Moore (Eds.), *Emerging Methods in Environmental Design and Planning*, MIT Press, Cambridge, p. 21-37.

Goel, V., 1995, *Sketches of Thought*, MIT Press, Cambridge MA.

Goldschmidt, G., 1990, "Linkography: Assessing Design Productivity." *Tenth European Meeting on Cybernetics and Systems Research.*

Goldschmidt, G., 1992, "Serial Sketching: Visual Problem Solving in Designing." *Cybernetics and Systems: An International Journal* 23, p. 191-219.

Gross, M. D. and E. Y.-L. Do, 2000, "Drawing on the Back of an Envelope: a framework for interacting with application programs by freehand drawing." *Computers and Graphics* 24(6), p. 835-849.

He, L., E. Sanocki, A. Gupta, et al., 1999, "Auto-summarization of audio-video presentations", In *ACM Multimedia 1999*, ACM SIGCHI, p. 489-498.

McFadzean, J., 1999, "Computational Sketch Analyser (CSA): Extending the Boundaries of Knowlwedge in CAAD", In *eCAADe'99: 17th International Conference on education in Computer Aided Architectural Design Europe*, The University of Liverpool, UK., p. 503-510.

McFadzean, J., N. Cross and J. Johnson, 1999, *Notation and Cognition in Conceptual Sketching.* (VR'99) Visual and Spatial Reasoning in Design, MIT, Cambridge, USA.

Moran, T. P., L. Palen, S. Harrison, et al., 1997, "I'll get that off the audio": A case study of salvaging multimedia meeting records", In *CHI'97 Conference on Human Factors in Computer Systems*, ACM, Atlanta,GA, p. 202-209.

Newell, A., 1968, "On the analysis of human problem solving protocols", In J. C. Gardin and B. Jaulin (Eds.), *Calcul et formalisation dans les sciences de l'homme*, Centre National de la Recherche Scientific, Paris, p. 146-185.

Newell, A., & Simon, H.A., 1972, *Human problem solving*, Prentice-Hall., Englewood Cliffs, NJ.

Oviatt, S. and P. Cohen, 2000, "Multimodal Interfaces That Process What Comes Naturally." *Communications of the ACM* 43(3), p. 45-53.

Schön, D., 1985, *the Design Studio*, RIBA, London.

Stifelman, L. J., 1996, "Augmenting Real-World Objects: A Paper-Based Audio Notebook", In *Human Factors in Computing (CHI) '96*, ACM, p. 199-200.

Wilson, T. D., 1994, "The Proper Protocol: Validity and Completeness of Verbal Reports." *Psychological Science: a journal of the American Psychological Society / APS.* 5(5 (Sep 01)), p. 249-252.

Programming and Assisted Sketching

Graphic and Parametric Integration in Architectural Design

Pierre P. Leclercq
LEMA, Laboratoire d'Etudes Méthodologiques Architecturales, University of Liège, Belgium

Key words: Design , architectural models, implicit knowledge management, sketch interface.

Abstract: In this paper, we present our latest research related to the concept of a sketch-interface. After describing our vision of computer assisted design and the conditions necessary for its effective implementation, an original data model is presented, which covers different levels of representation and is grounded in a database of implicit information. We then describe our software prototype, which exploits the potentials of the digital sketch, in order to demonstrate how our ideas are pertinent and the feasibility of three kinds of applications. In particular, we argue in favor of using an architectural software program in relation to the sketch within the same computer assisted environment at an early stage in the design process.

1. INTRODUCTION

Most software programs currently used in architectural practices inadequately cover the phases of architectural design. Historically speaking, they are above all intended for the production phases. That is, once the goals of the design have been almost entirely defined, these tools allow users to express how this production can be carried out (plans and construction materials) and their probable performance levels (cost, thermal behavior and stability). However, at this stage, everything has been decided. Indeed, the object has already been designed, yet the designer has received no assistance as far as initial decisions are concerned, and no support in terms of decision making strategies. Moreover, how could current software tools for architectural production possibly aid the reflections of a designer of an inhabited space? Most of these applications supply only a representation of

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 15–31.
© 2001 *Kluwer Academic Publishers.*

the building's structural elements: the basis of their data model is mediocre, consisting only of the walls, floors and other technical or architectural features. In order to assist designers effectively in their creative work, it is necessary to use the same concepts and levels of representation as they do.

1.1 Conditions for Assistance

We believe that any software package conceived to assist architectural design must have two main characteristics. On the one hand, its model must be centered on the architectural spaces being designed, on the other, it must enable the user to combine different levels of abstraction.

The goal of the first condition is only to ensure that the foundations of the knowledge manipulated by the computer correspond to the concepts used by the designer. Clearly, the spaces created by the designer are above all functional, that is places destined to house several specific activities, which require particular spatial, physical and environmental conditions (e.g., appropriate temperature, air quality, acoustic characteristics, colors, lights). Any tool that does not integrate these concepts in its data structure cannot, in our opinion, pretend to play the role of an architectural design assistant, since it does not use the same model as the architect to represent its main object.

The second condition is linked to how well the concepts being manipulated are defined: because they are being designed, they become more or less abstract or concrete during the planning process. For example, the functions of a future building are located and arranged in units that are rather vague and incomplete in the beginning. Then, little by little, they crystallize into a geometric form that becomes more and more precise as the architectural plans are progressively drawn up (Figure 1).

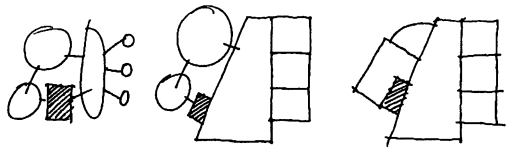


Figure 1. The evolution of an architectural sketch

Any software assistance tool must be capable of processing real architectural knowledge established using the same concepts of space and internal environment as the designer. At the same time, this software tool must effectively manage this knowledge with differentiated levels of information.

1.2 Assistance Media

Beyond this content, the mode of interaction between the designers and the tool assisting them must meet certain criteria to which we have already drawn attention through our discussion of architectural sketches (Leclercq, 1996). These criteria are inherent in the use of the sketch, which is the working method favored by professional architects. Though imprecise, vague, incomplete, and limited in its means of expression, the sketch is unquestionably the most popular mode of expression in the architectural design process. It is a means of simulation and a malleable raw material, which also creates a visible trace of the creative thinking process. In this way, the sketch can be considered as a reflection of a designer's cognitive work. It is a form of feedback for the designer: the model laid out on paper is not neutral, it reacts, strikes back, revolts against or conforms to the commands of the designer, who attempts to adapt it to the constraints of his or her project. In our model, the sketch is the means by which an important part of the designer's work can be captured.

1.3 Skills

Based on the analysis of 1600 professional sketch operations, we have already argued that it is possible to capture the main message contained in an architect's drawing. (Mathus, 1994). However, this potential is dependent upon two conditions:

1. There must be a common means of graphical expression that can be shared by all the designers concerned. In the area under consideration, we have observed that regardless of education (artistic, engineering), architectural style (functional, symbolic, organic, classic), and experience, all architects use a common, shared means of representation. It is important to note that our observations concern descriptive working sketches of the type exchanged between designers in a professional context. We are not considering artistic sketches intended for customers or for presenting the project to the public or to a jury, for example.
2. There must also be a large quantity of common implicit information, for within the architectural sketch the designer implements the figuration of basic concepts. Its economy of means makes it usable and preferable at an early stage of design, but at the same time limit its contribution to the management of secondary information from the designer. The designer considers this information as a body of implicit data, "which goes without saying," and enables the designer's collaborators to interpret effectively the global information summarized in the sketch.

We must add a third condition that any software package intended to assist design through sketches must respect. With its typical shortcomings of imprecision and incompleteness, the sketch represents a multi-level representational space. Some lines precisely denote the fine geometry of an architectural space that is already well defined in the designer's head. However, in the same drawing, a blob may appear: a vague, unclosed unit, recalling the presence of a space that has not been precisely defined, but that has already been characterized in terms of its function and the approximate area required (see Figure 1).
Yet it is exactly this lack of rigor inherent in the free-hand sketch, which seems like a fault when endeavoring to computerize the process, that makes the sketch such an interesting tool. In effect, free-hand drawing is the only means of expression that affords creative freedom. The level of abstraction it can attain guarantees access to:

- the spatial solutions, whether formal or functional, that the designer imagines while drawing them;
- the graphic ideas that emerge, whether intentionally or spontaneously, on the paper from the hand of the designer.

2. THE PROPOSED ARCHITECTURAL MODEL

We propose an architectural model that meets the three requirements elicited above, and which is thus capable of processing and interpreting a free-hand architectural representation. I shall now describe the basic principles of this architectural information model, about which you can find further information in (Leclercq, 1994).

2.1 Levels of Abstraction

Based on Rasmussen's research on the states of human knowledge (Rasmussen, 1990), we propose a three-level model of knowledge, represented schematically in Figure 2.

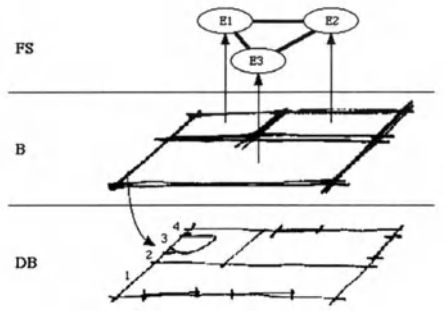


Figure 2. The three levels of knowledge in architectural design.

1. The first level of data representation can be called the " boundaries" (B). These stem from the graphic lines drawn by the designer in a working drawing , and represent the directly capturable traces of his or her thoughts. This representation is made up of architectural spaces, of inhabited areas, but is never represented as such. The designer establishes the boundaries, but at this stage, quite often it matters little to him or her how they will be made. What counts for now is the expression of spaces, the internal environments they are to contain, and the types of activities to take place within them.
2. In our model the spaces are concepts that have been deduced: they are generated based on the relational arrangements of the boundaries which cross or come together graphically. The second level is thus more abstract, and can be identified as that `which accomodates` space, associated with its functions and articulated in terms of its adjacency to other spaces. Independent of any metric or geometric constraint, this functional space (FS) network represents the essence of the plan in its relational and topological embodiment. It is therefore an architectural form of expression, which can be subjected to various forms of evaluation, such as a check concerning the building's accessibility. This form of architectural representation has a reciprocal relationship with the standard architectural representation by means of a plan, of which the sketch is an example. The sketch, therefore, represents an abstract expression of the architectural object being designed, in which a blob would fit in naturally.
3. On the other hand, at a more advanced stage when plans are more concrete, there is the level of detailed boundaries (DB), at which the lines representing the frontiers become more specialized. Characterized as doors, supporting walls or non-supporting internal walls, for example, the detailed boundaries are broken down into product models, the only way in which they are proposed in common CAD software programs. Therefore, this level leads toward a specialization of the elements in a

sketch, theoretically corresponding to the definitions of technical components found in CAD production software. Standard wall, column, beam, or concrete slab are identified and referenced at this stage using product catalogues, intended for take offs or other estimates: data which are extremely useful in the project phase that follows the design phase itself.

Beyond the fact that our model supplies this triple representation of the architectural object being designed, its main advantages are that the three levels are simultaneously produced and permanently linked.

- From the functional-space level, the designer can access a database of concrete cases and be assisted by functional or topological similarities in other designs.
- From the detailed boundaries level the designer can arrange the project at an advanced level of definition enabling direct access to the production phase of the project.
- By linking the topological diagram of spaces and the definitions of construction technologies at the level of detailed boundaries, certain evaluation factors until now inaccessible at the sketch phase become available to the designer, such as the estimation of energy performance levels and construction costs (as we shall see later in the presentation of the prototype).

2.2 Implicit Information

As I mentioned earlier, a sketch only reproduces the significant elements of the architectural project that it represents. It is not cluttered with obvious information that is assumed to be known or which will only be determined during the elaboration of the project. In effect, at this stage designers are not obliged or often do not wish to specify the semantic contents of everything they draw simply to reinforce their current design. In fact, they use a substratum of implicit data that we have described as three superposed layers (Figure 3).

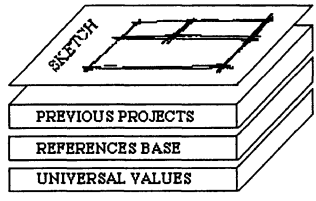


Figure 3. Layers of implicit information at the base of a sketch being drawn.

1. The first layer is a designer's own set of references, made up of previous designs, in which the designer or collaborators find everything that constitutes this designer's habits or personal preferences (e.g., usual ceiling height or sill height).
2. The second, more general layer covers the rules of good practice agreed upon by the community of professional architects to which the designer belongs. It includes the common rules, standards, and practices that any designer must observe in relation to national legislation, regional culture, or even local climate.
3. The third layer consists of universal references, such as material density values or thermal conduction coefficients for example.

Grounded in this collection of implicit data, the quickly sketched architectural drawing can be made more precise by an interpreting agent which, whenever necessary, performs searches in this hierarchical information base, from which the information needed to complete the architectural representation is taken. Thus organized, both in terms of prehension (abstraction/concretizing), and in terms of implicit information, our model is able to accommodate the sketched representation of an architectural object being designed. To continue my talk in more concrete terms, I shall now present our software prototype to demonstrate the feasibility of our propositions and how they operate.

3. THE DEMONSTRATION PROTOTYPE

The demonstration prototype, called EsQUIsE, is a geometrical interpreter of descriptive architectural sketches. For the time being the software packages only includes drawing procedures and lacks procedures for manipulating the objects once they have been drawn, and is therefore limited to the capture and interpretation of rough architectural sketches, and not yet to conceptual sketches. The sketch editing operations will be inspired by existing and very effective programs, such as Ideator (Kolli, Stuyver, Hennessey, Delft University of Technology, ND).

It is developed in a symbolic language, Common Lisp, and works using a Wacom digital sketchpad on all MacOS and Windows NT platforms.

It is made up of four basic modules (see Figure 4). First a graphic signal processing module, which synthesizes in real time the lines being drawn. Then there is a text recognition and semantic attribution feature for the captions in order to identify the function of each space. Finally, there are two procedures, carried out successively, to compose the spaces and deduce the topological relations in the architectural project being conceived.

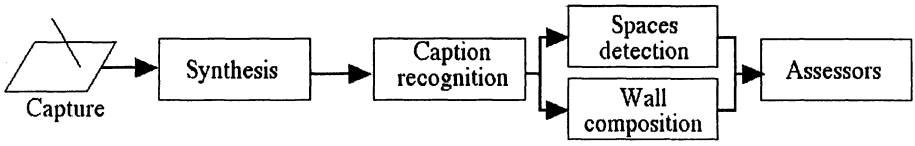


Figure 4. Procedural diagram.

The resulting architectural representation can then be sent to the usual software programs for assessing building performances. We shall now examine the role of each module in the order in which they come into action.

3.1 The capture module

The first module carries out the capture and synthesis of the lines drawn on the digital sketchpad. These can be in two colors, used to indicate the level of transparency of the walls: opaque or glazed. There is a third color available for non-significant additions to the sketch, for example shaded areas or the inclusion of furniture. (In time, we intend for this interface to evolve towards the concept of an "absent interface", to recreate the elementary, but immensely powerful, conditions of the paper sketchpad).
This capture module works at an acquisition frequency of at least 100 points per second, and processes a stream of information, whose size must imperatively be reduced in order for the process to be carried out in real time. A process of synthesis, made up of a set of successive filters, distills out the crucial information. The difficulty here is to reduce as much as possible the size of the representation of each line in terms of memory, while at the same time conserving a sufficiently reliable graphic synthesis of the original outline. These algorithms, whose objectives are obviously contradictory, are now capable of calculating a reliable synthesis of a line, in real time, using the equivalent of 15 % of the initial coordinate flow transmitted by the digital sketchpad.

3.2 The caption extraction module

Before the representations of the spaces are composed, a second module identifies and recognizes the functions of the spaces by means of the captions included in the sketch. These captions are necessary for establishing the architectural model described above, to be composed on the basis of the workings of the sketched plan. The technique, used in EsQUIsE since 1997, is based on the possibilities provided by the line synthesis module described above. Each line is decomposed into successive segments, whose orientations are coded in octants. Each line is thus translated in to a chain of concatenated codes, which can easily be compared with typical codes stored

in a character base. We should mention that Park and Gero uses the same technique for the categorisation of shapes in its work on the application of the genetic approach to the composition of architectural plans (Park & Gero, 2000).

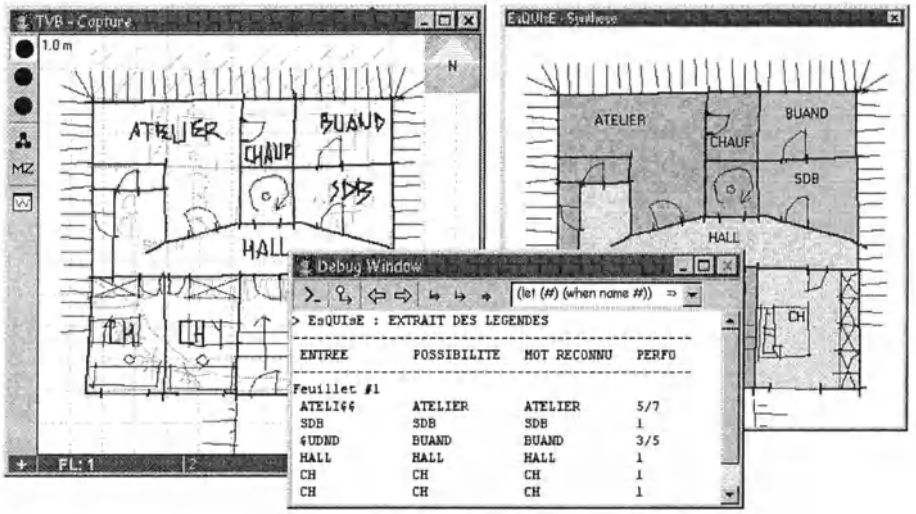


Figure 5. The original sketch, drawn on the digital sketchpad, the extraction of the captions, and the synthesized sketch.

This character recognition technique gives access to word recognition by means of a specific dictionary, which, in keeping with our proposal (figure 3), is located in the second layer of implicit data. Because of this limited dictionary, caption identification is very effective, since it is stable and gives performance levels equivalent to human performances. Thanks to this caption identification module, the prototype is able to name the different functional-spaces, and to link them to the characteristics usually associated with such spaces, such as the recommended temperature or noise level, which are also extracted from the database of implicit information.

3.3 Outline determination

By examining the points of contact of the synthesized lines, including the boundaries EsQUIsE can deduce the spaces demarcated by these lines.
These implementation procedures are fairly standard in the field of traditional computer graphics, where they are used in the techniques of outline determination in Boolean operations applied to coplanar polygons. The difficulty we encounter in our case is the imprecision inherent in an architectural sketch. In particular, we carried out a great deal of work on how

to define the ends of the hand-drawn line, which are always imperfect due to being interrupted, unfinished or overlapping. We therefore defined new procedures, called fuzzy computer graphics, able to continue or interrupt the dynamic movement of the line by processing error and proximity coefficients.

By overcoming this imprecision - though without correcting it, since the creative power of the sketch depends entirely on its unfinished nature – EsQUIsE can compose the second level of the architectural model, the more abstract level of functional-spaces. In addition, it can provide the characteristics of the boundaries between the functional-spaces. The size and orientation of each wall are easily located in the sketch. These metric data are then complemented by technological information, also from the implicit information database. Without any involvement of the designer, an interpreting agent itself selects the most appropriate type of construction technology. This selection is carried out by means of the internal environments separated off by each wall. These internal environments are identified from the functions determined by the caption recognition module. For example, between a bedroom and its adjacent bathroom, EsQUIsE chooses a wall that is sufficiently thick to provide soundproofing, and a wall covering that is impermeable to water vapor.

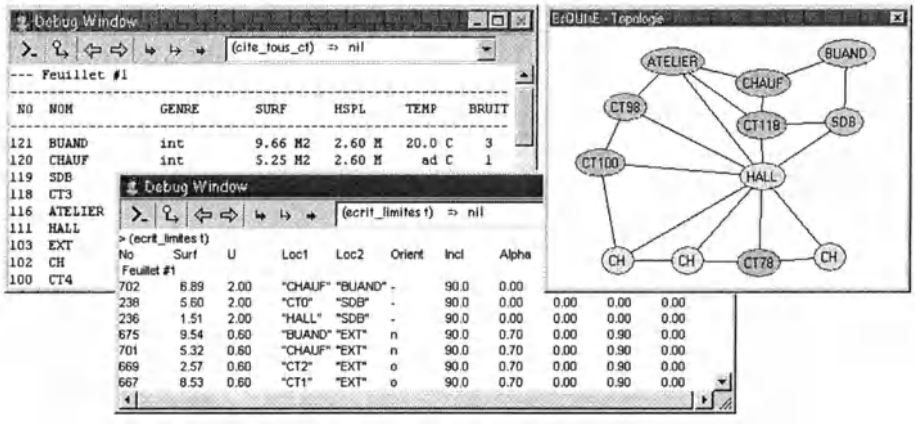


Figure 6. List and topology of the spaces and walls, together with deduced characteristics.

4. APPLICATIONS

The first application of EsQUIsE consists in generating, still in real time, a three-dimensional model of the layout corresponding to the sketch that has been drawn. Figure 7 gives a visual impression of the virtual scene available to the designer along with his or her digital diagram. This model is made up

in Quickdraw3D, under MacOS, and can be manipulated as if it were a cardboard model. It is built up directly from the lines drawn in the sketchpad, whose 2D sketches are taken from their respective pages. The designer can examine this virtual object as he or she wishes, from the outside, by rotating it, or from the inside, wandering through it so as to perceive its spatial qualities. Any modification to the various components of the sketch is, of course, automatically reflected in changes to the model.

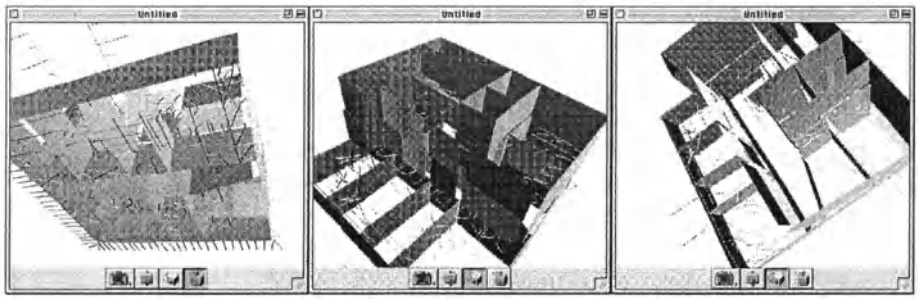
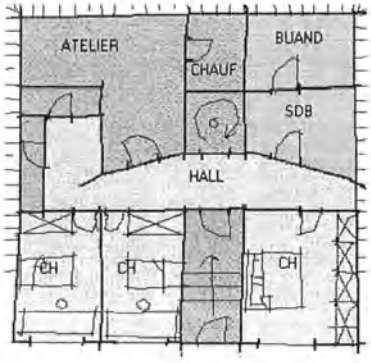
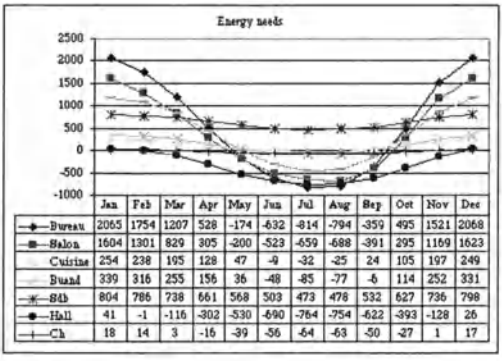


Figure 7. Manipulation of the virtual model

The second application of EsQUIsE consists of an evaluation of the energy needs of the architectural project. With a complete architectural representation of the sketched building now at its disposal, our prototype can submit it to a variety of different assessment processes. The results of these estimates each correspond to a forecast performance for the future building, which makes them very useful indicators for the design process. One application, called MZS, tested since 1998 downstream of EsQUIsE, is a conventional multi-zone evaluation module for the energy needs of a building.





	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	Nov	Dec
Bureau	2065	1754	1207	528	-174	-632	-814	-794	-359	495	1521	2068
Salon	1604	1301	829	305	-200	-523	-659	-688	-391	295	1169	1623
Cuisine	254	238	195	128	47	-9	-32	-25	24	105	197	249
Buand	339	316	255	156	36	-48	-85	-77	-6	114	252	331
Sdb	804	786	738	661	568	503	473	478	532	627	736	798
Hall	41	-1	-116	-302	-530	-690	-764	-754	-622	-393	-128	26
Ch	18	14	3	-16	-39	-56	-64	-63	-50	-27	1	17

Figure 8. – The energy needs of the sketched project are evaluated in an instant.

From the surface area and orientation of each window, MZS estimates the amount of sunshine likely to fall at each point and balances out the heating or air-conditioning requirements of the entire building, seen as a three-dimensional network of points. This method of calculating energy needs has been known for a long time, but its application in the context presented here demonstrates the advantage of our machine-user interface based on the use of the architectural sketch. Without the EsQUIsE prototype, the assessment shown in Figure 8 would require a great deal of tedious measurement and encoding of the building. With EsQUIsE, the building's energy performances are provided in a matter of a few seconds after the last line of the sketch has been drawn.

4.1 The Parametric Approach to Architectural Design

4.1.1 Definition of Constraints and Objectives

When the design process is examined (Gero and Neill, 1998), (Leclercq and Locus, 2000), we notice that before even drafting a first sketch to solve a given problem, the designer should first work out his or her domain of exploration. In Figure 9, this first phase is identified as that of the Definition of Constraints and Objectives.

Five sources of constraints, objectives, indications and inspiration can generally be identified: the client, the user, the site, the administrative authorities and the design team itself.

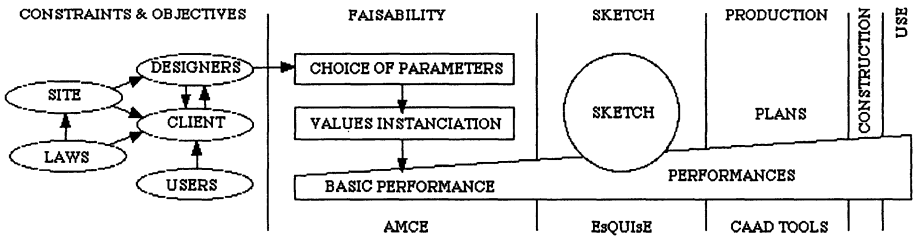


Figure 9. Structure of the parametric approach and a sketch.

Generally, constraints are integrated through a series of phases of negotiation, during which clients and designers refine their points of view and mutually expand their understanding of the problem. In this way, they progressively determine the outlines of the sub-space of potential solutions. In doing this, they prepare for the second stage in the design process by designating the determining parameters of the future project.

4.1.2 Feasibility Assessment

The second phase in the programming consists in assessing the existence and then the size of the sub-space of solutions. This step is very often neglected, especially in small or medium-size projects. But the issue of posing the question of feasibility appears to be crucially important at this stage in the process. There are two reasons for this:

- it forces the clients and designers to cover the full extent of the problem, and especially to characterize it by identifying its critical parameters;
- it enables them to progress towards the drafting of a solution with at least a minimal guarantee of a successful outcome.

However, it is clear that many architects – sometimes through incompetence, often through their eagerness to get started on the creative phase of their work – postpone this initial feasibility assessment step until after they have drawn their sketch. In so doing, they work with the graphic expression of their proposed solution, convincing the client and themselves that the process is operating smoothly. The feasibility assessment phase is tacked on after this cursory draft project Any feasibility test that turns out negative leads the designers into a phase of corrective engineering, during which the appropriateness of the proposed architectural solution is no longer called into question.

The next step in the feasibility assessment phase consists in assigning a value for each parameter that has been judged to be critical in the previous step. Any building can be defined according to four main sets of criteria: architectural expression, constructive principles, technologies of the building envelope, and functionalities. Instantiation of the related parameters, even when carried out fairly approximately, gives, by means of various simulations, a rough estimate of the crucial performance levels of the future building. Assessing these performance levels provides the answer to two questions concerning the above-mentioned sources: (1) thus defined, is the project feasible (i.e. will it meet all the external constraints)? And (2) is the project acceptable (i.e. will it satisfy all the internal objectives)?

With this enhanced understanding of the theoretical performance levels of their building, the designers can now start on the truly creative part of their work. Guided and supported by the predefined boundaries of the sub-space of solutions, their ideas can be more effectively formalized by means of the sketch than if the preliminary feasibility study had not taken place.

4.1.3 Illustration

To illustrate the potential benefits of this complementarity between the parametric approach and the architectural sketch, we shall examine the first

results of studies carried out at the LEMA-ULg by our colleagues J-M. Hauglustaine and S. Azar, within the framework of a research project commissioned by EDF (Electricité de France), entitled AMCE, Aided Method for the Conception of the building Envelope (Hauglustaine, 2000). This research team developed a software application that uses the Parametric Approach and connected it up with EsQUIsE. In concrete terms, the AMCE parametric interface presents a set of different tables in which each "source" enters his or her constraints. Figure 10 gives an example of the definition of the simplified geometry of a building as seen, on the one hand, by the future owner and, on the other hand, by the architect. It is clear that the level of detail chosen by the owner differs very obviously from that used by the architect.

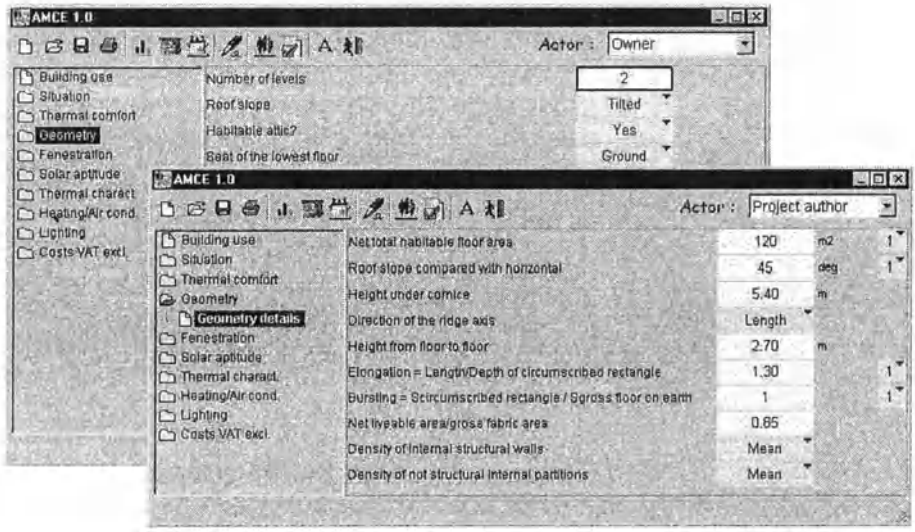


Figure 10. Geometrical constraints, as expressed first by the owner, then by the designer.

The introduction of parameters is not limited merely to assigning numerical values to them. The interface also enables the user to employ descriptive terms, couched in natural language, in order to specify the margin of possible evolution of the project. For example, the architect can express the wish that the number of walls should be "medium", thus indicating that he or she will neither be working on an open-plan design nor on a plan with too many juxtaposed internal spaces. This preliminary definition of the parameters covers all the normal domains of architectural design: use, location, thermal comfort, geometry, wall technology, heating, ventilation, lighting and economic considerations (62 parameters are now covered by the application). In the case of AMCE these are all related to EDF's particular requirements regarding the external envelope of the building. At this stage,

the designer can embark upon the creative process using the digital sketchpad provided by EsQUIsE (see Figures 5, 6 and 7). At any moment during the design process, he or she can check the performance levels of his or her model according to the interpretation derived from it by the software package. In the context of AMCE, these performance levels are presented with regard to the preferences of all the "sources" at the same time.

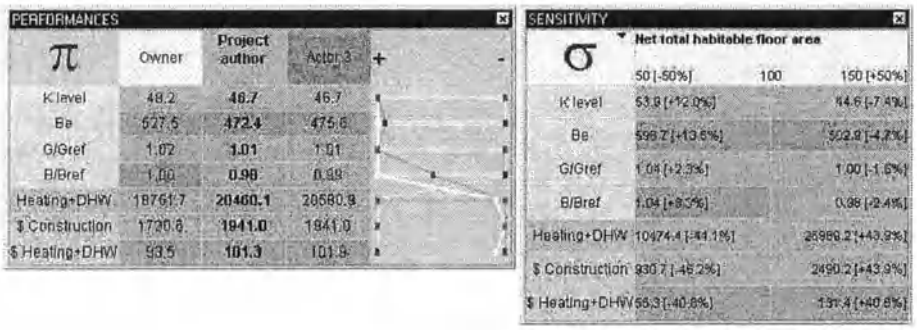


Figure 11.
Performance levels expressed with regard to each "source" and of performance sensitivity.

Figure 11 (left), for example, gives the regulation energy performance levels (K, Be, G, and B) of the first sketch, together with building and operating costs. It shows that the building sketched by the project designer (middle column) gives better energy performance levels than those based on the interpretation of the requirements of the future owner. On the other hand, the construction and operating costs would be considerably higher than the original targets. In this example, therefore, the sketch needs to be modified in order to make it correspond better with the desired performance levels.
Finally, a results analysis module evaluates the sensitivity of the results with regard to any of the parameters considered in the programming phase. Figure 11 (right), for example, shows the impact on the different performance levels of a modification to the net habitable floor area, leaving all other parameters unchanged. In particular, it shows that a reduction in habitable area would make it possible to satisfy the economic constraints without excessively marring the thermal characteristics of the building envelope.
Progressively calibrated in this way by the designer, the architectural draft project can quickly be presented to the different parties involved, with the guarantee that the future performance levels of the building project will, as far as possible, be satisfactory to them all.

5. CONCLUSION AND FUTURE PROSPECTS

I have just described two complementary approaches of interest to the first stage of the architectural design process: (1) a parametric approach, which concerns the expression of constraints, objectives, and the feasibility of designing a future building (the AMCE research program); (2) a graphic approach, which because of the flexibility of its fuzzy expression opens up possibilities for creative exploration while respecting the working methodology proper to architects (the EsQUIsE program). I have explained the interest of linking these two approaches and showed, through the results obtained from the two software programs, how the quality of the designer's work can be improved, and in addition the increased satisfaction for the project sponsor and the other individuals taking part in the creation of the building. A designer can work in a solution field that is *a priori* compatible with that of his or her counterparts. A designer's proposition can be completed with greater freedom thanks to the sketch-interface software tool, which reproduces typical free-hand working conditions and assists in project elaboration through cross-references from a pertinent implicit knowledge database, generated from the parametric definition of constraints and objectives.

Developments in the EsQUIsE research program scheduled for 2001 concern first of all the extension of the notion of a sketch to include the generation of three dimensional volumes from working drawings other than superposed plans: sections and elevations must be integrated in order to perfect the architectural representation. Second, we will attempt to apply our data model to the implicit formulation of requests during design with a database of specific cases. Faced with the numerous propositions of architectural case data bases that are unusable, because they lack an adequate interface, we believe that EsQUIsE can provide the key for implementing the theories that have been developed in this area since the 90's.

The AMCE research program will close in May 2001 with an application test in an architectural firm. An estimate of its potential contribution will be established based on the use of this prototype in the professional world. Unavailable today, the results from these tests will be announced in July during the talks at CAAD Futures 2001.

By formalizing the architectural sketch in a useable and concrete manner, and thus displacing it from paper to a virtual medium, we will open up a new field of use for the sketch. Thanks to their capacity to recognize the roles and relationships of and between the components in an architect's sketch, our prototypes provide a design environment that gives active assistance at an early stage of architectural design, thus assuring that the process will be a coherent and creative one.

6. REFERENCES

Do, E. and M.D. Gross : 1996, "Drawing as a mean to design reasoning", Visual representation, reasoning and interaction in design, Artificial intelligence in design '96 (AID'96), Standford University.

Gero J. and T. Neill: 1998, "An approach to the analysis of design protocols", *Design Studies* 19, 21-61.

Gross M.D. and E. Do : 1996, "Ambiguous intentions - A paper-like interface for creative design", Proceedings of Ninth annual symposium for user interface software technology (UIST'96), p. 183-192, M. Brown and R. Rao (Eds), Seattle.

Hauglustaine, J-M.: 2000, "Multicriteria and multiactors aspects of an interactive tool aiding to sketch the building enveloppe during the first stage of the architectural design", 52th meeting of the Working Group : Multicriteria Aid for Decisions, Vilnius, Lithuania.

Hauglustaine, J-M.: 1999, "Aid to the conception of the building enveloppe, during the first stage of the design", International Energy Agency Annex 32, Integral Building Enveloppe Performance Assesment, Report STA-F-99/1.

Lebahar, J-CH.: 1983, "Le dessin d'architecte - Simulation graphique et réduction d'incertitude", Paranthèses Edition, Presses Universitaires de France, Paris, France.

Leclercq, P.: 2000, "Programmation et esquisse assistées pour l'intégration des contraintes en avant-projet d'architecture", International Conference on Systems Engineering and Information & Communication Technology, NimesTIC 2000, Nimes, France

Leclercq P. and M. Locus: 2000, "Analysis of architectural design processes. Between formal and functional", poster of the *6th conference on Artificial Intelligence in Design (AID'00),* WIT, Worcester MA, USA.

Leclercq, P.: 1998, "Application d'un outil d'interprétation d'esquisses architecturales", La lettre de l'IA, number 135, Complex Systems, Intelligent Systems & Interfaces, Nîmes, France.

Leclercq, P.: 1997, "Programme EsQUIsE: acquisition et interprétation de croquis d'architecture", La Lettre de l'IA, number 123, Man-Machine interaction, Montpellier, France.

Leclercq, P.: 1996, "From an architectural sketch to its semantic representation", International Journal of Construction Information Technology, Vol 4, n°3, University of Salford, UK.

Leclercq, P.: 1994, "Environnement de conception architecturale pré-intégrée. Eléments d'une plate-forme d'assistance basée sur une représentation sémantique", doctoral thesis in applied sciences, Faculty of Applied Sciences, LEMA, Liège University, Belgium.

Mathus, P.: 1994, "Analyse d'esquisses architecturales", thesis in architect-engineering, LEMA, Liège University, Belgium.

Navez, J-CH.: 1996, "Dynamique de l'esquisse", thesis in architect-engineering, LEMA, Liège University, Belgium.

Park S.H. and J. S. Gero: 1999, "Qualitative representation and Reasoning about shapes", in J.S. Gero and B. Tversky (eds), *Visual and Spatial reasoning in Design* (VR'99), Key Centre of Design Computing and Cognition, University of Sydney, Australia, 55-68.

Park S.H. and J. S. Gero: 2000, "Categorisation of shapes using shape features", in J.S. Gero (ed), *Artificial Intelligence in Design* (AID'00), Kluwer Academic Publishers, Dordrecht.

Rasmussen, J.: 1990, "Mental models and the control of action in complex environments", Mental models and human-computer interaction 1. Ackerman D. and Tauber M.J. (Eds). Elsevier Science Publisher B.V., Holland.

Gesture Modelling

Using Video to Capture Freehand Modeling Commands

Mark D. Gross and Ariel J. Kemp
Design Machine Group, Department of Architecture, University of Washington

Key words: three-dimensional interface, video, gesture

Abstract: Desktop video combined with gesture recognition can be used to build powerful and easy to use interfaces for three dimensional modelling. We have built a demonstration prototype of such a system. The paper describes our video capture and gesture recognition scheme and illustrates its use in some simple examples.

1. INTRODUCTION

The Gesture Modelling project aims to provide a design environment for generating, editing, and viewing three dimensional computer graphics models using freehand gestures in space. We wish to enable designers to sketch models in the air, without the restrictions of traditional model-building, traditional sculpture, or conventional CAD modelling, but with the advantages that each of these methods have to offer.

Designers of three-dimensional artefacts (architects, but also mechanical, civil, and industrial engineers) frequently work with models constructed using Computer-Aided Design (CAD) programs. Most CAD programs use a Window-Indicator-Menu-Pointer (WIMP) interface to control the creation, viewing and modification of 3D forms. The designer views and operates on the artefact in orthographic, isometric, and perspective projections. Although designers have become accustomed to this way of working, which derives from traditional paper-based practice, working on a three-dimensional artefact through a two-dimensional interface has some limitations. That is why, in traditional practice, designers often develop a physical working model in addition to two-dimensional projections. The designer can add and

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 33–46.
© 2001 *Kluwer Academic Publishers.*

remove pieces to the physical model, point to certain elements or identify directions and dimensions in three-space. These operations are clumsier in two dimensions, even with an isometric or perspective projection.

For these reasons we seek to develop interface techniques for interacting with three dimensional design models that transcend the two-dimensional plane. In working with a physical model designers use three dimensional hand gestures to point out features, change the orientation of the model, add and remove model elements. The Gesture Modelling project trades on designers' experience with three-dimensional gesture, aiming to extend the kinds of operations that are possible in reference to a physical model to take advantage of the capabilities of three dimensional computer graphics. In short, we would like to support informal creation, viewing, and manipulation of three dimensional designs using hand gestures.

Using inexpensive desktop video to obtain three dimensional co-ordinates of the designer's hands, Gesture Modelling enables the designer to trace forms in space by directly manipulating three dimensional images. The designer maintains a sense of three dimensionality by interacting spatially with the computer graphics model. The paper describes our first steps in constructing a demonstration prototype Gesture Modelling system, written in C using Microsoft's Direct3D API. We outline our current physical arrangement of cameras and screen, the image processing routines used to identify hand gestures, and the mapping between hand gesture and modelling operations that Gesture Modelling uses.

2. RELATED WORK

Early systems for capturing hand gestures, including "Put That There" (Bolt, 1980), (Foley, 1987) depended on instrumented gloves. For example, VPL's Dataglove (Zimmerman, Lanier, Blanchard et al., 1987) measured finger bending using flex sensors and hand position using Polhemus position sensors.

Interacting with computer-aided design systems through gesture has been an attractive idea for some time. Fifteen years ago, the Maestro project (Nemeth, 1984) proposed a CAD work station that employed gesture as well as natural language to mediate the process of designing. Maestro also proposed a language of gestures for CAD modeling, including gestures for mimicking organic and manufacturing processes, manipulating and shaping forms, and defining space. Since then, several researchers in Computer Aided Architectural Design have experimented with using VR hardware to develop immersive design environments (Donath, 1999; Pratini, 1999; Regenbrecht, 2000)

While sensing gloves are becoming lighter weight, less expensive, and wireless, the advent of inexpensive desktop video and advances in machine vision techniques now make possible a new class of uninstrumented solutions to sensing gestures. A good, if somewhat dated, review of efforts to sense gesture using video can be found in (Huang and Pavlovic, 1995).

The approach we follow is similar to that of Segen and Kumar(Segen, 1993; Segen, 1998) and Bimber (Bimber, 1999).

3. DESKTOP CONFIGURATIONS

3.1 The gesture space and the model space

Although the combination desktop VR display and video input allows for an informal and direct interaction with the three dimensional model space, it has some drawbacks. One drawback is that when the input model space (where the designer gestures) is directly in front of the display, then the designer's hands are always in front of the 3D display. If the display is a CRT or flat panel, then the designer's hands obscure parts of the displayed 3D model; if the display is projected on a flat surface, then parts of the model are projected on the designer's hands, which cast shadows on the flat surface.

Figure 1. One of our Gesture Modelling configurations: camera and gesture space beneath the desk, with lights, projector and mirror displaying the model image on the desk top.

To address this drawback, in our current system we separate the input space and the model space. In one configuration, (figure 1), a perspective view of the 3D model is projected down onto the desk surface. A video camera is attached to the bottom surface of the desk top looking down, and the designer gestures beneath the table. The display projected on the desktop

includes both the image of the designer's hand(s) and the virtual computer graphic model that the designer is working on (figure 2). In this way, the designer's hands can be partly obscured by parts of the model.

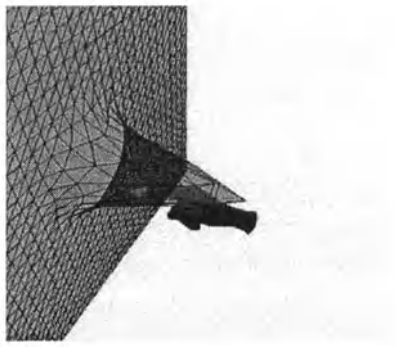


Figure 2. The display of the model space composed with the designer's hand.

3.2 Calculating depth with a single camera

Using a single camera, the system can estimate the z-co-ordinate (position in depth) of the designer's hand. The farther the hand is from the camera, the smaller the hand appears. By counting pixels in the hand profile, it is possible to obtain acceptable z-co-ordinate data. Figure 3 shows this technique. However, the number of pixels in the hand image depends on the orientation and configuration of the hand, not just its distance from the camera. Refinement of the depth-estimation algorithm would have to take these factors into account. If the frame capture rate is fast enough, we can use knowledge about the hand's most recent position and configuration to constrain the depth-estimate. Matching the hand profile against stored images of known configurations may also be useful in getting accurate depth estimates.

Figure 3. We can estimate the hand's depth position by counting pixels in the profile image

We also explored using a second camera, positioned so that it views the model space orthogonal to the first camera (Figure 4). Both cameras are inexpensive desktop models that communicate with the computer via USB.

A second video stream of the model space would help resolve depth ambiguity, as well as provide additional shape information about the hand configuration. However, accessing two USB cameras through the Microsoft's Direct3D libraries under Windows 98 and Windows 2000 caused unreasonably slow frame rates. Rather than debug this, we chose to postpone exploring the two-camera approach for a later phase of the project.

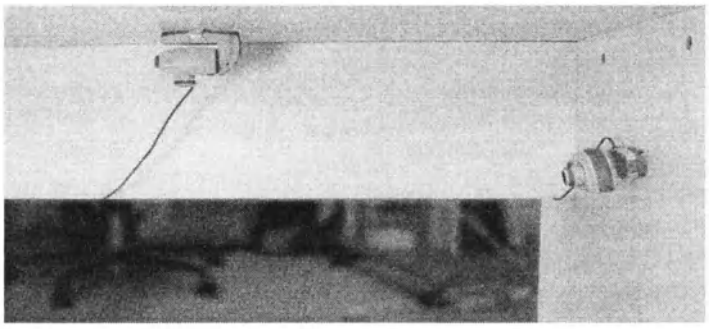


Figure 4. A second camera positioned orthogonal to the first

A problem with the configuration in figure 1 is that the designer must make hand gestures lower than is comfortable (around knee-height in a sitting position) to get the hand far enough away from the camera. To address this problem we experimented with using a mirror or two to extend the optical path, allowing the camera to be placed in a different location, for example, looking up at a mirror mounted on the bottom surface of the desktop. However, it is difficult to find positions for the camera and mirror such that the camera sees only the reflected image of the hand; that is, the hand does not come between the camera and the mirror, nor does the camera see both the hand and its mirror image. We tried mounting the camera on the floor, looking up at the gesture space. Although this resolves the distance-from-the-camera problem, it introduces two new difficulties because the camera sees the hands from the opposite side than the designer. Mounting the camera on the floor inverts the relation between the hand's z-co-ordinate and the distance-from-camera; the pixel-count method of depth estimating must be modified to account for this inversion. A more serious problem is that the camera sees the back of the designer's gesture while the designer would expect to see it from the front. This might be overcome by recognising the gesture profile and replacing the image inserted into the model with a stored view from above, or perhaps simply by flipping the image.

4. GESTURE RECOGNITION

Our approach uses pattern recognition techniques to identify the designer's hand configuration. The number of hand configurations is limited, so it may not be difficult to distinguish them even from a relatively low-resolution image. Also, the human hand can only move through possible gestures in certain ways; which constrains the recognition problem.

Wearing a black glove the designer gestures between the camera and a white background. The physical scene is lit by a diffuse light source (a luxo lamp shaded with white paper) located behind the camera, as well as a diffuse room light (a standard fluorescent light) located above the white background. This diffuse light source reduces shadows cast by the first light, which make the vision task more difficult.

The vision task consists of extracting from the image the following information:

a) the gesture of the hand, matching an item of a set of known gestures, or a 'none' output indicating no match is satisfactory
b) spatial information that will serve to generate forms, such as fingertip positions.

As the system is presently implemented the user's palm must remain oriented within a plane normal to the optical axis of the camera. Extended fingers must also lie along this plane.

4.1 Computation on input images

Input from the camera consists of a 160 by 120 RGB bitmap image (figure 5a). This is thresholded at a given intensity level, currently hard-coded in the system. (In the next phase of the project, we will compute the threshold intensity level from the first flat region after the primary peak of the intensity histogram.) The primary peak corresponds to the dark, 'glove' pixels. The histogram and the threshold level is computed only for the first frame of the sequence. The same threshold level is used for the remainder of the frames in the session. This eliminates unnecessary per-frame computation that would slow down the system. We may add to this a strategy to eliminate darker skin tones, which can become part of this primary histogram peak. Alternatively, the black glove can be worn with a bright long sleeve.

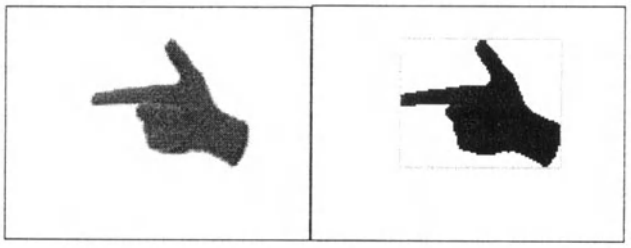


Figure 5 (a) raw camera image; (b) thresholded image with bounding box.

Pixels that are less than the threshold value are replaced by white (maximum intensity) pixels; those that are greater are replaced by black (minimum intensity) pixels. The result of this thresholding is the black silhouette of the gloved hand against a white background. A bounding box is obtained inscribing all the black pixels in the image (figure 5b). We aim to recognise the largest single continuous black object (blob) in the image. Only the region inside the bounding box is used for the remainder of computation.

Next, we compare the image inside the bounding box with a set of pre-stored low-resolution images, or templates. The problem of matching a template to an image is well-researched; however, a novel, computationally light approach is introduced here since the task is quite specific—recognising a *configuration* of a single known object that is guaranteed to reside within a given bounding box.

This approach consists of producing templates that are spatial maps of image features that can be compared pixel by pixel with the box-bounded camera image for fast per-frame performance. This contrasts with algorithms that extract features such as edges, segments, or lines from the unknown (camera) image, processes that would slow down the per-frame rate substantially. The templates are currently produced by a human operator. However, future work will focus on generating these templates automatically during a 'training' or 'calibration' period. This is important because the templates are produced from a series of real camera images of the a single user's hand; the system currently performs best when operated by that user. As the calibration need not perform in real time, we may use more sophisticated, computationally intensive methods to pre-compute the templates.

4.2 Template images

The template images and the matching strategy are next described. A good match with one of the template images constitutes a recognised gesture; the template has attached additional spatial information used in form synthesis.

Each template image stores information that serves to differentiate one gesture from another. An example of such a template is shown in figure 6.

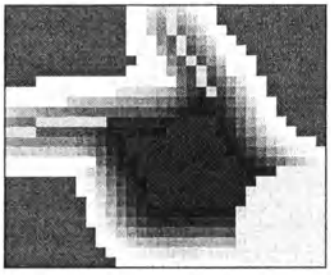


Figure 6. A template image for a 'gun' gesture. The image shown here is magnified

This template image was created by hand from a camera image manipulated with the Photoshop program as follows: The raw camera image is thresholded and cropped at a box bounding the gloved hand pixels. The resolution is decreased using bicubic resampling (presently to a size in which the greatest dimension is 32 pixels) and subsequently a Gaussian blur is applied with a mask of radius 1.5 pixels. Areas that remain white are replaced by red pixels. Areas that remain black within the palm are replaced by blue pixels. A region of grey pixels remains unmodified. Next, lines are drawn that indicate finger positions. As fingers are stick-like, they appear as black linear regions surrounded by the white background. Pixels in the centre of these black linear regions are indicated in green; the white regions around them are indicated in yellow. A composite of some templates stored by the system is shown in Figure 7. Notice that multiple rotations of the same gesture are represented. This eliminates unnecessary per-frame computation (and loss of accuracy due to re- or sub-sampling) that would be required for run-time rotation.

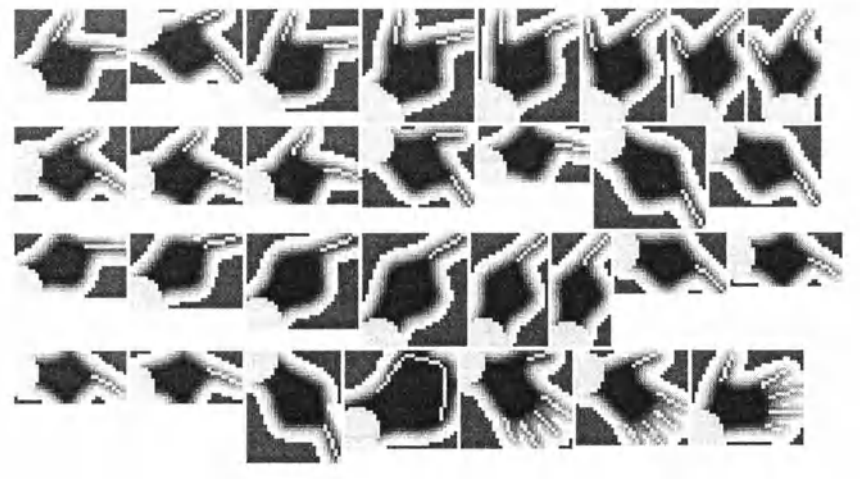


Figure 7. A composite of some templates stored by the system

4.3 Matching the templates:

The running program now compares each template image (**T**) to the unknown image. The unknown image (**U**) consists of the pixels contained within the bounding box. A scoring system determines the best match. First, the aspect ratios of **U** and **T** are compared. If they differ by more than a constant amount, **T** is 'discarded' and cannot be considered a match. (The next template image is considered, starting over the comparison process). Next, **U'** is obtained by subsampling **U** to match the dimensions of **T**. (This is not done explicitly; rather, pixels in **U** are accessed as if **U'** were subsampled). Red pixels in **T** are compared with the corresponding pixels in **U'**. If too few (a fixed fraction) of the corresponding pixels are white in **U'**, **T** is not a match. Then the matching score is incremented by an amount proportional to the percentage of pixels that do match. A similar process compares blue pixels in **T** with black pixels in **U'**. Then, green pixels in **T** are checked against the corresponding *un-subsampled* pixel in **U** and its eight neighbors. The score is incremented by an amount proportional to the number of pixels out of these sets of 9 that are black. A similar process is done for the green pixels in **T** and white 9-neighbors. Grey pixels in **T** are compared to pixels in **U**, and the score is incremented by how close the intensity values are. The grey area thus permits variations in the two images that can still produce a good match. Turquoise pixels in **T** aren't matched against pixels in **U** or **U'** because the system shouldn't compare wrist angles. Finally, the final score is thresholded to eliminate weak matches, and the top-ranked 3 templates are noted. From this list, the system selects the

gesture that matches the largest number of the last several frames' top-ranked gestures. This avoids gesture 'flicker' in case of a wrong match. An equivalent strategy would be to change the reported gesture only after it has been top- or near-top- ranked for *n* sequential frames.

4.4 Recognition performance

Shown below are images of the recogniser's 'vision' window. The coloured pixels represent the spatially corresponding, with respect to bounding boxes, matched template image. The program is run using Windows 2000 and a USB personal desktop camera:

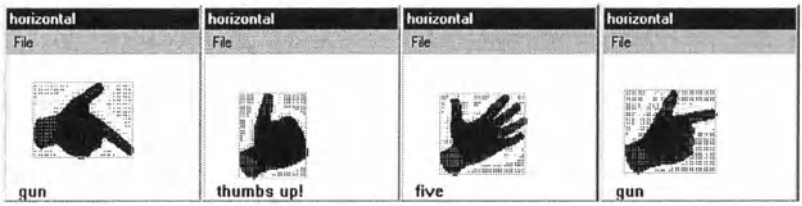


Figure 8 from left: 'gun' gesture, thumbs up!; 'five' and 'gun' with imperfect match

We have not subjected the system to rigorous quantitative testing, but the demonstration applications we have built perform well in real time. The system has been trained to recognise seven gestures: point, pinch, gun, thumbs up, fist, five (open palm), two (v-symbol). The system is fairly robust, as can be seen from figures 8-c and -d, where the template image and the camera image do not match as directly as they could, yet the proper gesture is reported. A benefit of matching the template image to the bounding-box of the gloved hand is that spatial information (fingertip position, etc) used to create forms is continuous, even though template images are not (for example, template images differ by rotations of 10 degrees).

Most gestures are recognised correctly, with the correct orientation and few non-gestures are reported as matches. In a test of the terrain modeling application (see below), for example, five users who had no experience with the system were immediately able to make recognized gestures. However, this system only employs three distinct gestures: (five, point, and fist). The biggest problem using the system is extraneous pixels from the user's sleeve or other parts of the body entering the camera's field of view, which confuses the recognizer.

Recognition templates are prepared in Photoshop and loaded into the system. The Gesture Modeling environment cannot be trained or customized for a different user, nor can new gestures be added without editing and recompiling code. It could be extended to train the system interactively to recognize new gestures. This would require building the image processing functions described in section 4.2 (Gaussian blur, bicubic resampling, clipping). Though not conceptually difficult, this remains a project for future work, as does the interactive binding of gestures to specific application commands.

5. APPLICATIONS

5.1 Form Generation

The first version of the Gesture modelling system generates a series of points in the model space as the designer traces a line in the gesture space. The points are represented in the model space by tiny tetrahedra, so that they are visible from any viewpoint (Figure 9). In this first version, the first dark pixel found in the video frame (scanning from left to right and top to bottom) was used to identify the (x y) co-ordinates for the new point. If the designer uses his/her index finger as a "brush" and is careful not to rotate his or her hand, this simple method of getting the co-ordinates works well. However, the designer must orient his or her hand so that the index finger is always the first thing the system sees in this scan pattern. We find this constraint unacceptable.

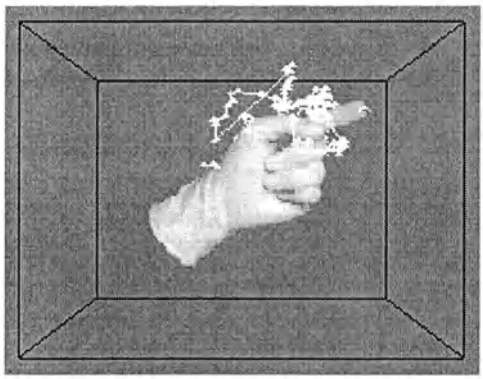


Figure 9. The system leaves small tetrahedron at the fingertip (first pixel scanning from left, top) using hand size to determine a Z co-ordinate

We considered painting the tip of the designer's index finger (or a latex glove) to find the red pixels in the image, and using this to specify the (x y) co-ordinates of the insertion point. We did not complete implementing this approach. Instead, we began to work on recognising the designer's hand gestures, which would offer a more general and less constrained interaction.

5.2 Mesh Distortion

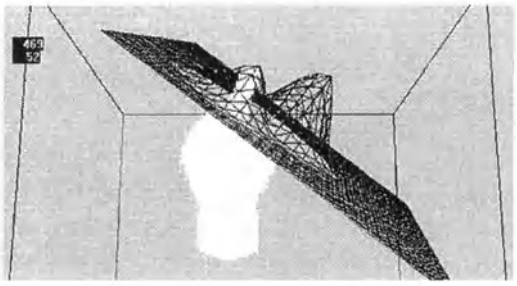


Figure 10. Creating a landform by deforming a mesh

Figure 10 shows how, using the 'point' hand gesture, the designer can deform a mesh to create a land form. The numbers at the left of the screen indicate the co-ordinates of the hand's position.

5.3 Selecting and moving objects

Figure 11 shows how, once a land form mesh has been created, the user can place objects on the mesh surface and on top of one another. Two rectangular solids have been placed on top of a hill in the land form. The designer is using the 'fist' gesture to grasp and carry one of the objects. In the next version of the software, we plan to run a simple hydrology model to simulate water runoff on this landform.

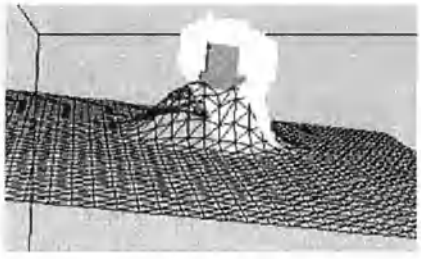


Figure 11. Selecting and moving objects

Figure 12 shows how, using the 'open palm' gesture, the designer selects one among a set of blocks. As the designer's hand passes over each block, it highlights in red to indicate that it is selected. If the designer then forms a fist, the selected object is grasped and can then be moved in three dimensions.

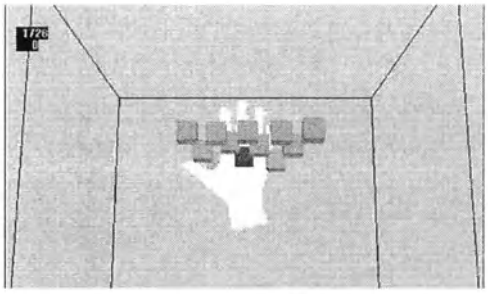


Figure 12. Selecting blocks to arrange them in a massing model

Figure 13 shows how the designer can make an arrangement of blocks, by grasping them as above and then placing them in three-space.

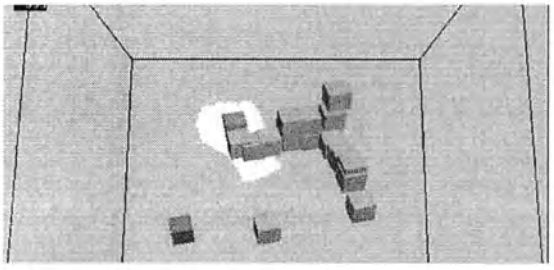


Figure 13. Creating a massing model by assembling blocks

6. FUTURE WORK

The present Gesture Modelling program is little more than a proof of concept demonstration and a platform for experimentation. We have used it to create several small demonstration systems, in which particular gestures are bound to particular operations. For example, in the terrain modeler, 'point' deforms the mesh; 'five' selects a block, and 'fist' moves a selected block for placement. However, these are specific examples, and we have not used the Gesture Modelling program to build a larger system, nor have we tested the system's recognition more than a handful of distinct gestures. We

have also not explored the use of dynamic gestures. These are immediate directions for further work.

7. ACKNOWLEDGEMENTS

We would like to thank the anonymous CAAD Futures reviewers for their helpful suggestions. This research was supported in part by the US National Science Foundation under Grant No. IIS-96-19856/IIS-00-96138. The views contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

Bimber, O., 1999, "Rudiments for a 3D freehand sketch based human-computer interface for immersive virtual environments", In *VRST 99*, ACM, London, p. 182-183.

Bolt, R., 1980, "'Put That There': Voice and Gesture at the Graphics Interface." *Computer Graphics* 14(3), p. 262-270.

Donath, D., 1999, "Using Immersive Virtual Reality Systems for Spatial Design in Architecture", In *AVOCAAD Second International Conference [AVOCAAD Conference Proceedings*, Brussels (Belgium), p. 307-318.

Foley, J. D., 1987, "Interfaces for Advanced Computing"." *Scientific American* 257(4 October 1987.), p. 126-135.

Huang, T. and Pavlovic, 1995, "Hand Gesture Modeling, Analysis and Synthesis", In *Proc. International Conference on Automatic Face and Gesture Recognition*, p. 73-79.

Nemeth, C., 1984, *Maestro: A Gesture Recognition System.* Institute of Design, Chicago, Illinois Institute of Technology.

Pratini, E., 1999, "Esboçando com Gestos: O Projeto 3D SketchMaker", In *III Congreso Iberoamericano de Grafico Digital [SIGRADI Conference Proceedings]*, Montevideo (Uruguay), p. 141-144.

Regenbrecht, H., Kruijff, E., Donath, D., Seichter, H. and Beetz, J., 2000, "VRAM - A Virtual Reality Aided Modeller", In *Promise and Reality: State of the Art versus State of Practice in Computing for the Design and Planning Process [18th eCAADe Conference Proceedings*, Weimar (Germany), p. 235-237.

Segen, J., 1993, "Controlling Computers with Gloveless Gestures", In *Proceedings Virtual Reality Systems*, New York City, p. 1993.

Segen, J., 1998, "Gesture VR: gesture interface to spatial reality", In *International Conference on Computer Graphics and Interactive Techniques; abstracts and applications*, ACM SIGGRAPH, p. 130.

Zimmerman, T. G., J. Lanier, C. Blanchard, et al., 1987, "A hand gesture interface", In *CHI+GI*, ACM, Toronto, p. 189-194.

Toward the integration of spatial and temporal information for Building Construction

Shen-Guan Shih and Wei-Lung Huang[1]
National Taiwan University of Science and Technology
[1]Easylines Building System Co. Ltd.

Key words: building construction, project modelling, representation

Abstract: Building construction is a kind of complex process that integrates activities regarding design, materials, personnel and equipments. Information systems that describe building construction need to represent both the spatial information of the building design and the temporal information of construction plan. We classify four levels of integration for spatiotemporal information in building construction. We consider that the classification is important for the guidance of our research for that it distinguishes levels of complexity and applicability of data models that integrates spatiotemporal information. A prototype system is developed and tested for providing us a means to gain more insights to the problem.

1. INTRODUCTION

Many construction projects follow the conventional designer/constructor sequence, in which the owner employs designers to prepare drawings and specifications for the constructor. Traditionally, in the design phase of a project the designer should not communicate with potential contractors to avoid collusion or conflict of interest. One of the major disadvantages of such designer/constructor sequence is the failure to revise design early enough to reduce construction time and cost. Decisions made at the beginning stages of a project life cycle have far greater influence than those made at later stages, as shown schematically in figure 1. Therefore, it is important that decision makers should obtain the expertise of professionals

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 47–60.
© 2001 *Kluwer Academic Publishers.*

to provide adequate planning and feasibility studies for design development and construction planning in early stages.

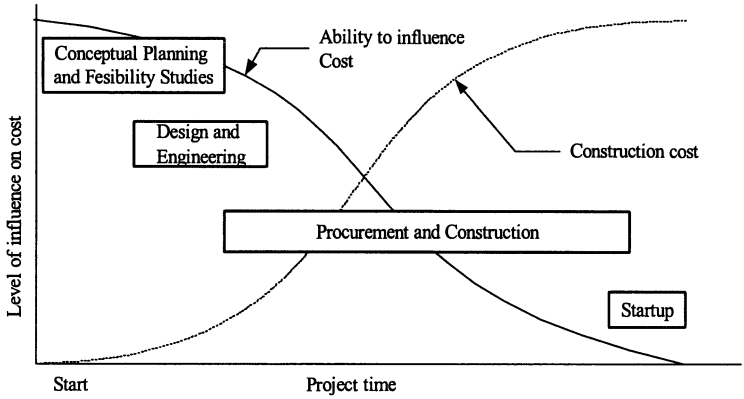


Figure 1. Ability to influence construction cost over time (redrew after Hendrickson, 1989)

Designers are concerned with the spatial status of the building that is to be built, while constructors emphasize on the timing and process of the construction. Drawing and specifications are the basic means of expressing the spatial status of the building. Powerful tools based on information technology have been developed to assist the manipulation and the integration of drawings and specification documents. The most often used representations of temporal information regarding construction processes are networks and charts. Computer applications for construction management have been made available since decades ago. However, what remains unavailable is the link between design and construction, which until now requires engineers to manually analyse the spatial and temporal information of the project in the process of design and construction planning.

The separation of spatial and temporal information is necessary and effective in regarding to the ease of manipulation and human comprehension, as well as for the application of conventional data processing techniques. Information technology has great advances both in terms of modelling systems that describe spatial information, as well as project management systems that manage the temporal information of planning and scheduling. Efforts heading for the integration of spatiotemporal information can also be found in previous works such as 4D CAD systems [Retik, Warszawski and Banai, 1994] [Mey and Heide, 1997][McKinney and Fisher, 1998], which add temporal attributes into building elements to describe the construction process of a building, and

temporal GIS systems [Story and Worboys, 1995][Langran, 1993][Al-Taha, and Barrera, 1990], which consider the dynamic changes of geographic environments along the axis of time.

This paper describes our efforts in putting together concepts and techniques in 4D CAD and in temporal GIS systems to define the scope and the structure of an integrated environment that would assist decision-making for the design and planning of a construction project, emphasizing the representation and analysis on relations between design and construction. More specifically, the focus of discussion is on the data model that enables the representation and the analysis of the interrelationships between the spatial, declarative information defined in drawings, and the temporal, procedural information that are consisted in construction plan and schedule. We propose that the integration of spatiotemporal information would be useful and feasible for supporting designers, project managers, and construction planners in the designing and planning of construction projects. A prototype that links a design modelling system, a database management system and a project management system has been developed and evaluated. With further development, the proposed model might as well extend its applicability into the construction stage of the project.

2. SPATIOTEMPORAL INFORMATION

Designers work on spatial specification of a project with drawings, models and other documents to define the geometry, the material and the structure of the design that is to be constructed. Constructors use networks and bar charts to represent construction processes and schedule, which constitute the temporal aspects of the project. The representations of spatial and temporal information in a project are different and separated, yet the relation between the design to be constructed and the process to construct is important. A good project management would rely on the ability to integrate and analyse both spatial and temporal information of the project to assist decision-making in design and construction plan. It is expected that with good management, a design with required performance would be constructed in a reasonable process, spending minimal cost and time. Information in a project can be classified into the three categories, which are design, process, and cost. The diagram in figure 2 indicates that the three categories are closely interacting. Design is defined mainly in spatial information; process is defined with temporal information, whereas cost is related to both temporal and spatial aspects of the project.

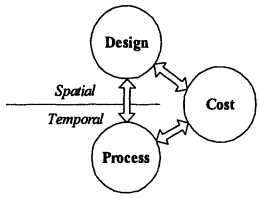


Figure 2. Categories of information in a construction project

Table 1 describes the contents of spatial information and temporal information in a construction project. Spatial information includes position and dimension; together with other attributes such as material and function define the spatial status of design components. A design can be regarded as a composition of primitive objects, each of which relates to others in some specific ways. Temporal information describes temporal aspects such as time and duration of activities, which are basic components that define construction processes. An activity may relate to other activities in regarding to procedural dependency, the use of equipments, materials, personnel and other resources. Spatial information may relate to temporal information in various ways. For example, a project may be divided into several phases of development. Different parts of the design might be in different stages in the life cycle of the project at a given time. Two construction activities may have spatial conflicts regarding working areas or spatial constraints, which are difficult to express in terms of process network and construction schedule. The use of equipment might be restricted by the spatial status of the site at some intermediate stages of the construction. Furthermore, the change of design might result in the change of form, quantity and material that may generate inconsistency that is difficult to foresee by analysing either design or construction schedule alone.

Table 1. Contents of spatial and temporal information in a construction project

Spatial information	Temporal information
Position	Time
Dimension	Duration
Component	Activity
Spatial relation/topology	Sequence and dependency

2.1 State, event and activity

There are various models for the representation of time and process. In this paper, we use the notion of state, event, and activity as elements to

define the temporal aspect in a construction process. A state is a specific set of values that defines the status of one or a set of objects. An event marks a specific position at the time axis for the change of states. An activity consists of a starting event, an ending event and a state that connects the two events. An event occurs at one position in temporal axis. An activity has a duration that is measured by the distance between temporal positions of its starting event and ending event. Therefore an activity would divide the time axis into three parts, which are the period before the activity, the period when the activity is being performed, and the period after it has been completed. Conventionally, with CPM method, a construction process consists of a network of activities, each of which can be related to other activities by sharing its starting or ending event with other activities. The network expresses the temporal sequence of those activities.

2.2 Relations between spatial and temporal information

Spatial information of a building can be expressed as a set of objects, each of which consists of a set of attributes that define the geometric and non-geometric properties of the object. An event of the construction process triggers the change of states of some objects. A design object relates to an activity if its state changes upon the starting and the ending events of the activity. Therefore, the evolution of an object is divided by a relating activity into three states, namely, the pre-activity state, the in-activity state, and the post-activity state. Each object can be related to one or more activities, and each activity can be related to one or more objects. With this definition, the state of an object remains unchanged within the period when a relating activity is in progress, unless there are interferences of other activities.

3. SPATIOTEMPORAL INTEGRATION

We consider four levels of integration of spatiotemporal information. Each level is progressively more difficult yet more powerful than its precedent level according to the order in the following sections.

3.1 Level 1: The recording and presentation of sequential changes of objects over time

This first level of integration for spatial and temporal information is to store the construction sequence in the geometric model, such as the 4D CAD model that can be found in some commercial applications such as

Microstation schedule simulator. It is similar to the undo function, with which a user can trace back and forth a history of operations. Every operation that changes the state of the system is regarded as an event that can be stored, backtracked and replayed according to the temporal sequence. The same mechanism can be used to record and replay events of a construction process to show how the spatial status is changed in an arbitrary stage of the construction. As what is shown in figure 3, every state change of each object in the temple front construction is recorded and linked with related construction activities. The states of objects can be shown graphically in a drawing to represent the spatial status of a given time in the construction process. At this level of integration, an object in the drawing can be associated with events that change its state. For example, objects C1, C2, C3 and C4 are associated with starting and ending events of activity *setup_column* and activity *paint_column.* Underneath is a list of events marked with time of occurrences and related objects.

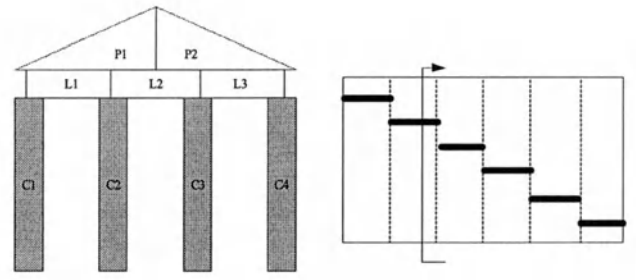


Figure 3. A presentation of the spatial status of a given time in the schedule

```
(1 (setup_column starts) C1 C2 C3 C4)
(1 (setup_column ends) C1 C2 C3 C4)
(2 (place_lintel starts) L1 L2 L3)
(2 (place_lintel ends) L1 L2 L3)
(3 (place_pediment starts) P1 P2)
(3 (place_pediment ends) P1 P2)
(4 (paint_pediment starts) p1 p2)
(4 (paint_pediment ends) p1 p2)
(5 (paint_lintel starts) L1 L2 L3)
(5 (paint_lintel ends) L1 L2 L3)
(6 (paint_column starts) C1 C2 C3 C4)
(6 (paint_column ends) C1 C2 C3 C4)
```

3.2 Level 2: The cross analysis of spatial and temporal information in construction process

In the conventional way of representing construction schedule and drawings, analysis that considers both spatial and temporal information requires engineers to manually interpret the relating documents such as construction schedule and drawings. The integration of spatiotemporal information may support automatic analysis of spatiotemporal information to perform the following tasks.

- Analyze the influence on construction schedule when a change of design is requested.
- What activities are active at a given time in a given area?
- Find all occurrences in the construction process where there are more than one active activity in the same area at the same time.
- Show the working areas of activities that would need to use the same kind of equipments.
- Estimate the desirable quantity of some materials in a given working area at a given time.

Using the temple front example shown in figure 3, the system would be able to answer questions such as,

1. What is the total weight that has to be lifted in activity *place_pediment*? And what is the maximum weight and volume of one single piece to be lifted?
2. What are the working areas of *place_pediment* and *place_lintel*? Does it need different installations of weight lifting equipments for the two activities?
3. How much paint do we need in day 5, day 6, and day 7? How do we refine the schedule to level the labor that is needed to paint the temple front?
4. How many workers can most efficiently do the job of *paint_pediment*, considering the amount of work and the size of working area?

And it may assists design decisions regarding how the construction pieces of the temple front can be divided to shorten the construction schedule, and to make the use of equipments and personnel more efficient. Integration at this level requires a CAD model that incorporates much information concerning construction time and cost, such as material and quantities in addition to geometric attributes such as location and shape.

3.3 Level 3: Automatic analysis of spatial relationships between construction activities

The activity network of a construction has to be planed according to the relationship among construction activities for the project. The relationship depends on the use of resources, equipments, spaces, as well as the states of related objects before a specific activity can be started and after it is completed. The integration of spatiotemporal information may enable automatic analysis of the interdependencies of construction activities according to the states of related objects and spatial conflicts in the use of equipments and working areas. Descriptions on initial conditions that are required to activate an activity can facilitate the analysis of spatial relationships between activities. Considering the temple front example, we may refine the construction process. For example, column C1 and C2 has to be set up before L1 can be placed on top of them. The starting event of (place_lintel L1) requires C1 and C2 to be in the post-activity state of (setup_column). The dependencies of activities can be explicitly defined in the desirable relationships of events and states.

```
(starts place_lintel L1):
       ((C1 (post setup_column))(C2 (post setup_column))
       (L1 (pre place_lintel)))
(starts place_lintel L2):
       ((C2 (post setup_column))(C3 (post setup_column))
       (L2 (pre place_lintel)))
(starts place_lintel L3):
       ((C3 (post setup_column))(C4 (post setup_column))
       (L3 (pre place_lintel)))
(starts place_pediment P1):
       ((L1 (post place_lintel))(L2 (post place_lintel))
       (P1 (pre place_pediment)))
(starts place_pediment P2):
       ((L2 (post place_lintel))(L3 (post place_lintel))
       (P2 (pre place_pediment)))
```

According to the list of dependencies, the process that build the structure of the temple front can be further refined as the network shown in figure 4.

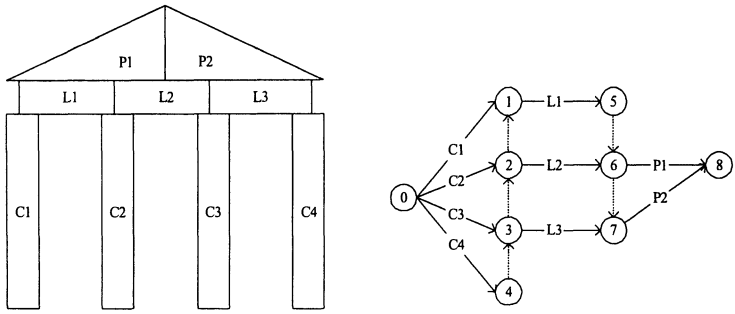


Figure 4. A construction plan for the temple front project

Furthermore, the schedule to carry out the construction plan must consider the use of equipment and labor. Suppose that the installation of the weight-lifting equipment is time consuming, and it is necessary to relocate it if the work is out of its reachable range, spatial analysis may help in the planning of construction schedule. In this case, if there is only one set of weight lifting equipment available, a refined sequence of activities can be C1 C2 L1 C3 L2 P1 C4 L3 P2.

3.4 Level 4: Graphical Simulation of the dynamic process of construction

Construction is a kind of dynamic process interlaced with interactions between personnel, material and equipments. It is not possible to foresee all possible consequences upon unexpected changes in the project. Simulation is a powerful means to supplement analysis to enhance the ability to manage a construction project. An ideal model for construction should provide necessary information to simulate the dynamic process of construction. It should go beyond merely visual animation of the process, but to simulate the execution of the construction plan upon possible situations that might result in unexpected consequences. We expect that a data model that describes the causal relationships between activities and state changes of object would form a computational basis for simulation. In addition, means of estimating how well an activity can be performed under a given condition has to be devised.

4. A PROTOTYPE

Based on the previous discussion regarding integration of spatiotemporal information, we have implemented a system to provide a test base. The framework of the system is shown in figure 5. The system is built upon the coordination among three modules, which are a modelling system (Microstation plus an architectural module called Triforma), a project management system (Microsoft Project) and a relational database management system (Microsoft Access).

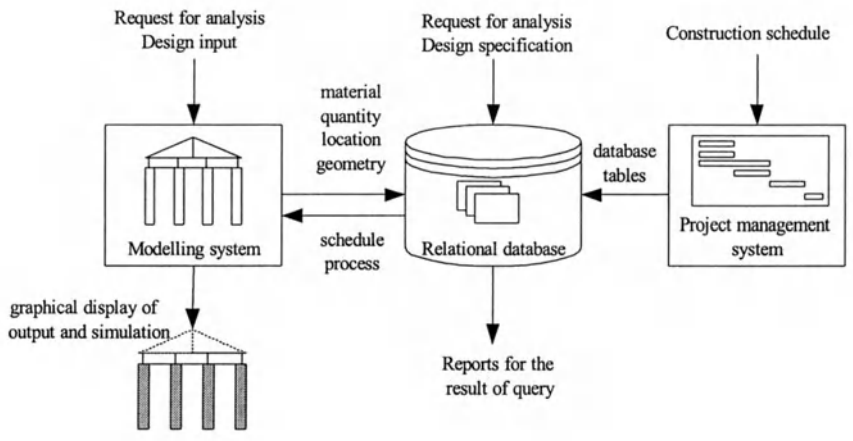


Figure 5. The framework of a prototype system

The modelling system is used as a front end for spatial information to model the building. Programs written in MDL (Microstation Developing Language) are used to provide graphical output, customized user interface, and necessary data processing for the connection to other modules. The project management system is used as a front-end for temporal information to input and to manipulate information regarding construction plan and schedule. The construction schedule is converted into tables and exported to the third module, a relational database, where spatial and temporal information are integrated. Objects in the building model is connected to the relational database using ODBC Data source and the database interface provided by Microstation. The connections between building elements and construction activities, as shown in figure 6 and 7, are made in the modelling system using a customized user interface. In our test case, elements of a 5 storey RC building are drawn as 3D objects in the modelling system (Microstation). Specifications of parts and components that are required in design and construction, such as steel, form, concrete, and finishing are

defined by the architectural extension (Triforma) of the modelling system. The construction schedule is input with the project management system (Microsoft project). The schedule is converted into tables and sent to the relational database that is created as an ODBC data source. Links between building elements and construction activities are done manually in the modelling system.

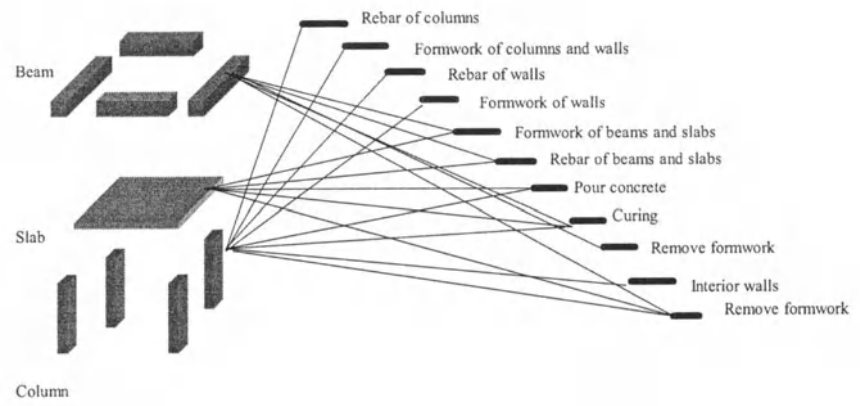


Figure 6. Links between drawing elements and construction activities

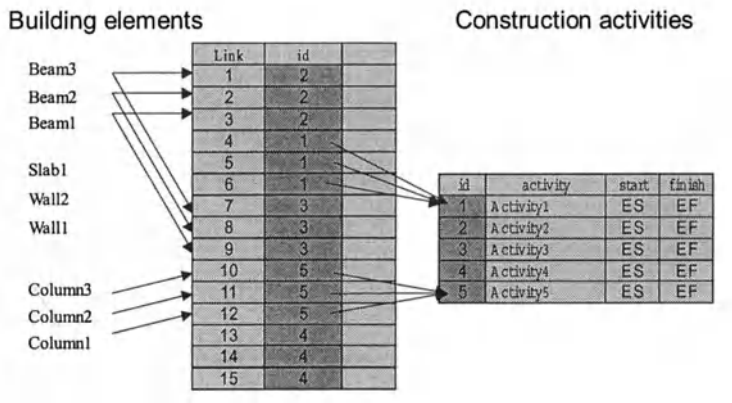


Figure 7. The many-to-many relation between building elements and activities

The analysis of the spatiotemporal information is performed through interactions between the modelling system and the database management system. In the modelling system, a request can be defined by indicating an

area, selecting building elements, or none-geometrical data through the interface. The system would combine spatial data with temporal data stored in the database to perform necessary analysis and provide either textual output or graphical output. The system can be used to trace and replay the construction process graphically, as shown in figure 8. The system is also able to perform cross analysis on spatiotemporal information as the level two integration described in section three of this paper. For example, the user may indicate an area and ask what activities would be performed at this place at a given time, and what are the types and quantities of materials and labours will be used? It is potentially capable of calculating all examples described in the second level of integration, although some of them are not implemented. The user can also modify the building model and the system will recalculate the quantity of materials and labour that are needed. Automatic adjustment of schedule would be feasible if combined with a program that estimates the needed duration of construction activities according to the quantities.

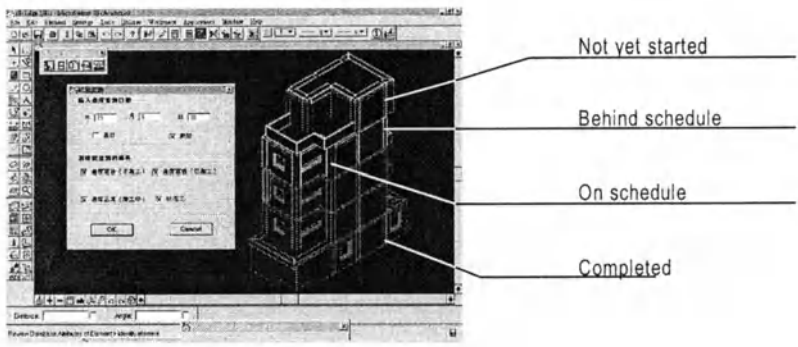


Figure 8. Graphical presentation of construction progress

5. CONCLUSION

The system is tested with a five storey RC building. With evaluation done by design and construction experts, conclusive remarks are made to direct the future development of research. First, the system is powerful, yet not practical in realistic projects. It requires too much work to prepare necessary data regarding the building model and construction process. It also requires tedious work to make links between every building element and its relating construction activities. It inherits some incompatibility regarding automatic calculation of material and cost to the construction profession in

Taiwan, where the system is developed and evaluated. A general opinion is that our future work should be directed toward decision-making support in the early phase of a project. It would be impractical if used for construction management, but could be helpful for design decision-making and construction planning.

Our current work has been heading to three objectives. First, it is important to provide higher levels of abstraction in the design model and construction schedule. In the early phase of a project, decision would be made based on more abstract data and rough estimation, rather than details in design and construction schedule. It has been tested on the integration of construction process with design study of building volume and site plan. Second, links between the building model and construction activities should be done more elegantly. For this purpose, a new module called spatial enquiry engine to select design elements and to provide spatial data is added to the system. The revised structure of the system is shown in figure 9. It is expected that the relation between construction activities and building elements can be pre-defined and stored together with construction plan. The links would be done upon the request of analysis on spatiotemporal information, instead of in the modelling of the building.

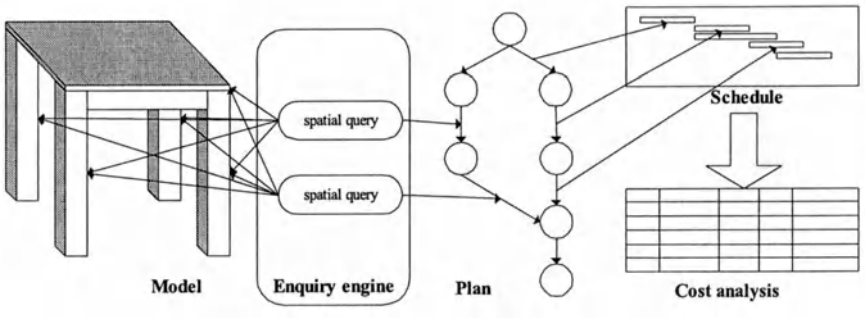


Figure 9. A revised system

Third, a new data model is being developed for the third level of integration, which is automatic analysis of spatial relationships between construction activities. Definitions of the initial state and final state of activities have to be added to the model, and the reasoning process has to be clarified.

6. ACKNOWLEDGEMENT

The National Science Council of Taiwan supported the research described in this paper. Authors would like to acknowledge Professor Lee Cheng-Sian and Mr. Hong, Guo-Yuan for their professional advices on the re-direction of our research.

7. REFERENCES

Al-Taha, K., and Barrera, K.R., "Temporal data and GIS: An overview", *Proceedings of GIS/LIS '90*. November 1990, 244-254.

Allen, E., G. Edwards, and Y. Bedard, "Qualitative Causal Modeling in Temporal GIS", *Spatial information theory: A theoretical basis for GIS*, A.U. Frank, W. Kuhn (Eds.) Springer, 1995.

Collier, E., M. Fischer, "Four-dimensional modeling in design and construction", CIFE Technical Report, NO. 101, Stanford University, Stanford, CA, February 1995.

Galton, A., "Towards a qualitative theory of movement", *Spatial information theory: A theoretical basis for GIS,* A.U. Frank, W. Kuhn (Eds.) Springer, 1995.

Langran, G., *Time in geographic information systems*, Taylor & Francis, London, Washington, DC 1993.

McKinney, K. and M. Fisher, "Generating, evaluating and visualizing construction schedules with CAD tools", *Automation in Construction* 7 (1998) 433-447.

Mey, M.G. and H. Heide, "Towards spatiotemporal planning: practicable analysis of day-to-day paths through space and time", *Environment and Planning B: Planning and Design* 1997, v24, p709-723.

Retik, A., A. Warszawski, A. banai, "The use of computer graphics as a scheduling tool", *Building Environment.* 25 (2) (1994) 133-142.

Story, P.A., and M.F. Worboys, "A design support environment for spatio-temporal database applications", *Spatial information theory: A theoretical basis for GIS*, A.U. Frank, W. Kuhn (Eds.) Springer, 1995.

Hendrickson, C. and T. Au, *Project management for construction*, Prentice-Hall, Inc. New Jersey, 1989.

Activity objects in CAD-programs for building design

A prototype program implementation

Anders Ekholm
Architectural and Building Design Methods, Lund Institute of Technology/Lund University

Key words: CAD, user activity modelling, building design, building product modelling, building brief development, space function program

Abstract: In the early stages of the building design process, during the programming and the proposal stages, both user activities and the building are in focus for the designer. In spite of this, today's CAD programs give no support for management of information about user activities in the building. This paper discusses the requirements on modelling user activities in the context of building design and presents a prototype software. The prototype is developed as an add-on to the architectural design software ArchiCAD.

1. INTRODUCTION

1.1 From drafting to modelling

The development of computer aided building design systems is currently shifting focus from drafting to modelling. With a model oriented CAD-program it is possible to develop an object based "product" model of the building. Building product models constitute a basis for achieving computer integrated construction and facility management, CIC/FM, processes. The shift from drafting oriented CAD to model oriented CAD enables new ways of managing and structuring design information (Eastman 1999).

CAD, computer aided design, has mainly been applied during the later stages of the design process when the building and its detailed design is in focus. In the early stages during the programming and the proposal stages both user activities and the building are in focus for the designer. Here, CAD has been used less frequently. One reason is the rigor that CAD-systems impose on the designer, as opposed to the quick and intuitive response of

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 61–74.
© 2001 *Kluwer Academic Publishers.*

manual drafting. Another reason, relevant for both manual and CAD drafting, is that a drawing is a static medium poorly suited to represent the dynamic user activities. A third reason, specifically relevant for model oriented CAD, is that today's model oriented CAD programs, e.g. ArchiCAD or Architectural Desktop, mainly have objects that represent the building. They do not support management of user activity information, since they have no activity objects to which one can attribute a user activity description.

1.2 Current research

Prototype information systems with explicit representation of user activities has to the present authors' knowledge only been developed by Eastman and Siabiris (1995). They identify "activity units" composed of furniture, equipment and activity area with the emphasis on spatial properties. Other approaches identify "functional units" (Flemming and Chien 1995) or "space units" (Carrara, Kalay and Novembri 1994). These represent functional requirements and generic spatial properties of the buildings spaces, and in that way indirectly represent the organisation units. The indirect approach is also used in "Alberti", a commercial space planning software by acadGRAPH. The present author has discussed user activity modelling in earlier papers, see e.g. (Ekholm and Fridqvist 1996). A later example is the work by Hendricx (2000). None of these have been realised as working CAD-prototypes.

1.3 Aim of this project

Modelling of activities and processes is an area in strong development, but has so far not been developed to suit the needs of building design and facility management. The aim of this project has been to develop a prototype software that can model user activities in the context of a CAD program for building design. The objective of this prototype is to show some of the potential functionality in design and facility management processes. Hopefully, it may also generate not anticipated ideas of possible applications to those experimenting with it.

The following section of this paper discusses the theoretical framework for modelling user activities, and the information of interest in building design. The third section presents the actual implementation and ends with a discussion about future development and applications.

2. MODELLING OF USER ACTIVITY SYSTEMS

2.1 Conceptual foundations for building design

2.1.1 The theoretical framework

In order to develop theories and methods within the field of design, it is necessary with a well structured generic theoretical framework including semantics, ontology, and epistemology. The theoretical framework used in this development project is based on Mario Bunge's Treatise on Basic Philosophy, specifically the parts on ontology and epistemology (Bunge 1977, 1979 and 1983).

Several concepts used in this section have been presented in earlier writings by the author. The interested reader is recommended e.g. (Ekholm 1987 and 1994, and Ekholm and Fridqvist 1996, 1998 and 2000).

2.1.2 A systems view on organisations

A basic concept in a description of reality is that of system. A *system* is a complex thing with bonding relations among its parts, it has composition, environment and structure, both intrinsic and extrinsic (Bunge 1979:8). A *process* is a sequence of events in a system. An *activity* is a goal-directed process. The terms 'process' or 'activity' may also be used to designate the system itself since it is a characteristic feature.

An *artefact* is a man-made or man-controlled system; it is made with a purpose to make certain activities possible. A human activity system that involves the use of artefacts is also called a *sociotechnical* system. *Work* is a specific kind of activity, it is a useful activity (Bunge 1979:197). A sociotechnical system engaged in some work activity is in management science called an "organisation" (Child 1984), "human activity system" (Checkland 1981), or "enterprise" (Bubenko 1993).

The organisations of modern society are complex sociotechnical systems organised in functional units composed of human individuals and equipment, including tools and machinery. An organisation has a spatial extension traditionally called *activity space*. The activity spaces are of different scale from the smallest, defined by the human body, tools and materials, to the space determined by the organisation as a whole.

To adopt a view, or aspect, on a system is to observe a specific set of properties. Of specific interest to design are the functional and compositional views. A functional view focuses on the system's relations to the environment and on parts that contribute to the system's function. A

compositional view of a system identifies the compositional parts from which it is assembled (Ekholm and Fridqvist 2000). See Figure 1.

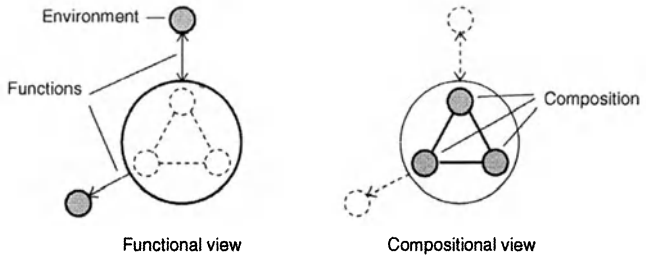


Figure 1: Functional and compositional views on systems

2.1.3 Design as problem solving

Design is an activity which aims at producing an artefact with specific required properties. Design may be seen as a problem solving process. A problem can be defined as a lack of knowledge about the properties of a thing or about how desirable properties of a thing can be achieved. Problem solving therefore is a knowledge acquisition process (Bunge 1983).

In general, a problem solving process proceeds in a sequence of steps. It starts with the recognition and definition of a problem in relation to some background knowledge, see also Figure 2 below. This is followed, first by synthesis, leading to a tentative solution, and then by analysis, investigating the proposed solution. The result of the analysis is added to the background knowledge. The cycle proceeds until a satisfactory solution has been developed.

Synthesis may be regarded as starting from a functional view on the design, while analysis starts from a compositional view. The synthesis question is: Which thing has these properties? And the analysis question is the inverse: Which properties does this thing have?

2.2 Building design

2.2.1 The role of CAD-programs

CAD-programs support the building design process in different ways. A traditional CAD-program for drafting allows the designer to document the geometrical properties of the design. A drawing supports both synthesis and analysis, however information capture from the drawing requires human visually based interpretation. A model oriented CAD-program allows the designer to document a much more complex conceptual model of the design

in the computer. This computer based model, in construction IT named product model, does not require human interpretation for information capture, but may be directly accessed by different application programs.

2.2.2 User activity information

User activity information is used throughout the construction and facility management processes. When an organisation is formed or changed, it may need a new or renewed building for accommodation. The process of acquiring a suitable building starts with a description of the organisation and its activities. The *activity description* is used as a basis for developing a *space function program* which defines requirements on the building's spaces. The following step includes development of a *building program*. The building program together with the activity description and the space program are used as a background for building design, but can also be used for building performance analysis during the facility management stage (Svensson et al 1999).

As an example, the activity information needed for space function programming is listed below. The list is based on Hales (1984:17), and (Akademiska Hus 2000).

General activity description
- General description concentrating on factors determining spaces and installations

Activity relationship information
- Process sequence
- Material exchange between activities
- Communication, personal or through media
- Spatial relations, visibility, audibility, supervision, security, shared resources and other relationships;

Activity attributes
- Activity area
- Dimensioning measurements
- Duration
- Noise
- Heat production

Person information
- Personnel data; skills, working hours

Equipment information
- Furniture, machinery and equipment; quantities, measurements
- Products and materials

Building information

- Building (and process) support; HVAC-requirements, fire rating, sound proofing, electricity
- Lighting; daylight/black-out
- Atmospheric pressure

2.2.3 User activity information in building design

User activities in the building are seldom explicitly presented in drawings, but mostly left to imagine by the actors. Object oriented modelling opens up the possibility to explicitly represent user activities. Despite of this, today's model oriented CAD programs, e.g. ArchiCAD or Architectural Desktop do not support management of user activity information, since they have no activity objects to which one can attribute a user activity description.

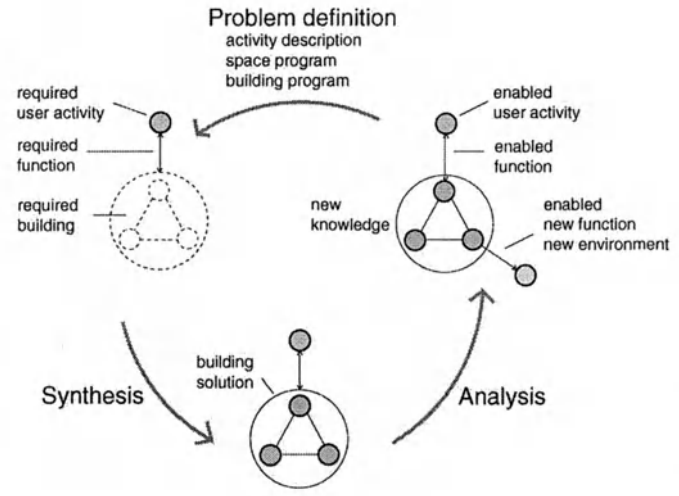


Figure 2. The building design process

A program that allows the development of a user activity description assists the problem definition work in the design process. See Figure 2. A model oriented CAD-program for building design also assists the designer's synthesis work, and allows the designer to document the decisions on the buildings properties. Building analysis includes analysis of technical performance, cost calculations as well as usability analysis. If the user activities are modelled in the same system, it would also support usability analysis. Usability analysis may be more or less dependent on human visual information capture, but since both the building and the activities are

represented as object models it should be possible to automate certain aspects of the analysis. After analysis, the results can be used as background knowledge in another cycle in the design process.

A conclusion is that CAD-systems by automating information management to different extent can support the three fundamental steps in the design process of problem definition, synthesis and analysis.

3. THE PROTOTYPE ACTIVITY ADD-ON

3.1 Add-ons

An add-on is a separate program that expands the functionality of a another program, and can only be run within this. The prototype program developed in this project is an add-on to ArchiCAD. It has its own user interface accessible from the interface of ArchiCAD (e.g.: new menus, dialogues, floating palettes, etc.). The API, Application Programmer Interface, for development of add-ons enables access to the inner processes and database of ArchiCAD. With these capabilities it is possible to enhance the basic ArchiCAD elements, but it is also possible to use independent tools and techniques and integrate them into the ArchiCAD environment.

3.2 Conceptual schema of the Activity Add-on

3.2.1 Entities

It has not been possible in this prototype development work to implement functions to manage all the information needed for space function programming as listed in section 2.2.2. However, a future implementation is both possible and desirable. The following section describes the actual implementation.

The basic entity of the Activity add-on is the Activity. It is based on a functional view on an organisation or part of an organisation. An Activity may have other activities as functional parts or itself be a functional part of other activities. Activities are composed of Person and Equipment. The constituent Person and Equipment may be determined for an Activity at any level in the "hierarchy".

Activities may have Name, Description, Duration, and Relations. There are four Relations that can be specifically shown: Visibility, Sound, Distance, and Adjacency. These may have values which, however, can only be described, functionality is not implemented.

A Person has Name and Description, it can only exist within an Activity.

Equipment may be composed of other Equipment. It may have Name and Description. An Equipment element can exist independently during the time period between the Activities in which it appear. Between Activities it has the same state as in the last Activity.

Compared with the information needed for activity description according to section 2.2.3 above, the prototype lacks a relation between activities and building spaces, as well as a resource entity.

The entities and their relations as implemented in the prototype are shown in the schema in Figure 3. The schema is presented in EXPRESS-G, a framework for graphical product model representation, based on the EXPRESS modelling language (Schenk and Wilson 1994).

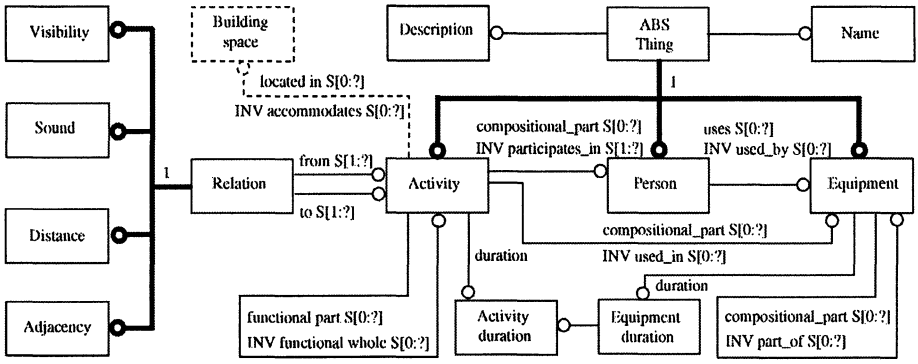


Figure 3: Conceptual schema of the Activity Add-on

3.2.2 The Object entity in ArchiCAD

The Activity Add-On can link Activity entities with Object entities in ArchiCAD. An Object entity is a library part within ArchiCAD, it can represent different things, e.g. people, furniture, specific building parts like kitchen fit-outs etc. However, the link can only be active if the Object entity has the "visible" variable within its parameters which can be edited by an advanced ArchiCAD user. Person and Equipment objects in the Activity Add-on are such edited ArchiCAD Objects.

Two new ArchiCAD Object entities have been developed for the Activity Add-on:

- The Activity Space Object which shows the spatial extension of the activity. It can be created, with the Fill→Activity Space transformation tool within the Activity Palette.
- The Activity Relation Object, which is handled by the add-on.

3.3 Functions of the Activity Add-on

3.3.1 Activity Menu

The user of the Activity Add-on manages an Activity System through the Activity menu in ArchiCAD. The Activity menu contains:

- Activity Settings; to switch on or off the Activity Settings palette
- Time Observation; to switch on or off the Time Observation palette
- Activity palette
- Save Report file

The Activity Add-on in the current implementation enables the user to determine and edit an Activity System. The main functions to configure the Activity System are handled from the Activity Settings palette, available as a

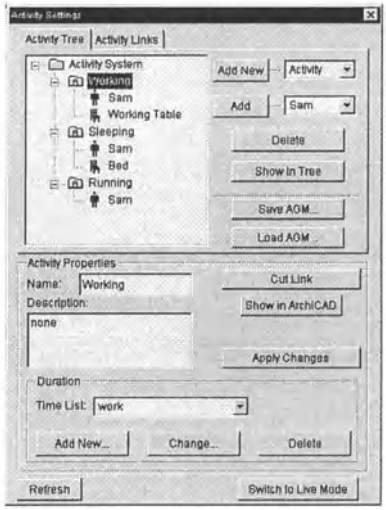


Figure 4: The Activity Tree dialogue box

dialogue box placed in the ArchiCAD window. The main functions are "Activity System", "Activity Links" and "Activity Properties".

An Activity System is made up of activities in a hierarchy of different levels. See Figure 4. Activities in each level are composed of Persons and Equipment. The first level of the hierarchy must be an Activity. The same Activity may only occur once within the Activity System, but Persons and Equipment can occur many times.

An Activity can be saved as an AGM, Activity Group Module. This module consists of the Activity and its parts as well as its composition of Persons and Equipment. An AGM can be saved in a library and be reused.

The relations between activities can be determined in the "Activity Links" dialogue box. The relations are: Visibility, No Visibility, Sound contact, No Sound contact, Connection, No Connection, Adjacency, and Distance. The links can be set at direction and certain grades of importance. The links are shown in a "relation stamp" on the screen.

The Activity Properties includes name, description and duration. The user can control the appearance of Activities and the related objects in time through the "Time Settings" dialogue box. An Activity can be set to be either periodical or happen once. See Figure 5.

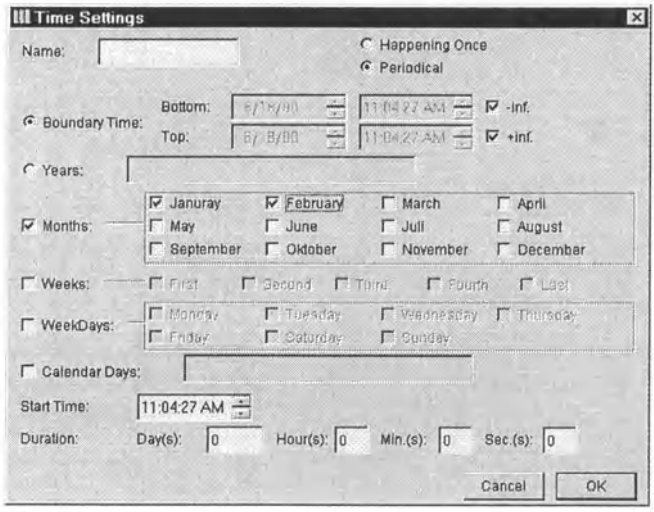


Figure 5: The "Time Settings" dialogue box

3.3.2 Activity Space definition

The user of the Activity Add-on may start the activity design work by defining a hatched area developed through some of the ArchiCAD geometry tools. Then, when the designer wants, it is possible to define the hatched area as an Activity Space using the Activity Space transformation tool. A similar function works for other ArchiCAD objects like Wall or Slab. This means that a designer initially can work with a geometrical representation, and at will determine whether the object shall be, for example, an activity, or a slab.

3.4 The School Test Case

The Activity Add-on has been applied to model a small school and its lessons. The different tools in the program have been used to define the School Activity System. An example of the design tool in use in the Edit

Mode is shown in Figure 6, where the Floor Plan and Activity Tree shows shows *Art* in class A, and *Computer* in class B.

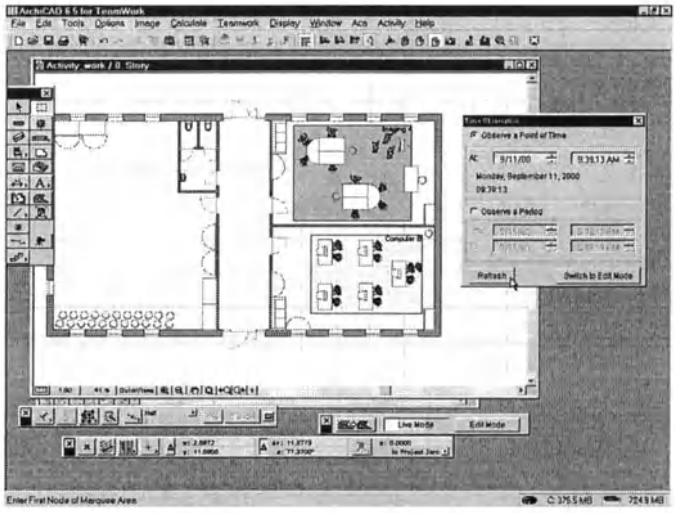


Figure 6: Art in class A, and Computer in class B, observed on a Monday morning

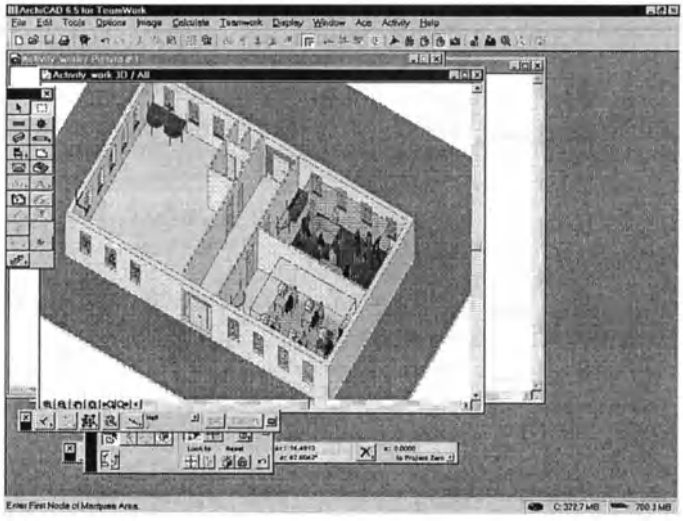


Figure 7: In the 3D mode also the extension in 3D of the Activity Spaces is visible

The activities are observed at 09.39:13 on Monday morning, September 11, 2000. In Figure 7 the same activities are shown in a 3D view. Also the extension in 3D of the Activity Spaces is visible.

3.5 Conclusions and future development possibilities

3.5.1 Conclusions

The Activity add-on to ArchiCAD strongly enhances the functionality of building design software in the problem definition and analysis phases of design. The integration of activity objects in software for building design opens new possibilities for building design methods development.

3.5.2 Further investigations

Several aspects should be further investigated. For example:

- Illustration of user activities
- Spatial lay-out design
- Temporal space use analysis
- Versatility analysis
- Space function programs
- Activity libraries
- Process modelling

Buildings are built to enable activities of different kind. During design or in facility management it is necessary to be able to *illustrate user activities* in 2D and 3D and how they are accommodated in the building. This functionality may be extended towards animated and interactive representations by sophisticated software and hardware. Traditional software for building design enables illustrations of people and equipment, but not as graphical representations of activity objects.

Methods for *spatial lay-out design,* wich coordinates spatial requirements of buildings and activities, may benefit from the availability of activity objects with relational and spatio-temporal attributes. Spatial allocation of an organisation can be supported by graphical representation tools like: adjacency matrices, flow-charts, blocking and stacking diagrams, graphic and net area displays, spatiotemporal diagrams etc. Such tools are computer implementable and could be part of computer aided organisational design programs. See e.g. (Kalay and Séquin 1998), Kumlin (1995), Hales (1984), Child (1983) and Muther (1973).

The time settings functionality of the Activity add-on clearly illustrates the possibilities to develop methods for *temporal spaces use analysis*. The use of a buildings' spaces is time dependent. The possibility to generate space lay-out plans for different times of the day gives a new insight in the possibilities to co-ordinate activities to achieve a more effective space use.

Methods for *analysing the versatility* of a building and its spaces are important both during building design and facility management. The versatility of a space is a measurement for its capacity to accommodate different activities. Flexibility, e.g. by movable partitions can be illustrated, by representing a partition as Equipment of an Activity System. As a background for methods development it would be of specific interest to analyse the SAR-methods developed in the 1970's by John Habraken and his colleagues (Habraken et al 1974).

Further program development should be made to allow the user to develop *space function programs* an important part of the problem definition work. Information about activities and required building properties are stated in the space function program, which is used both as a starting point for the building design process, and as a background for performance studies during facility management.

Activity systems together with their building requirements should be possible to store in a reusable form as *library objects*. One source of inspiration for developing and structuring such coordinated activity-environment objects would be the Pattern Language methodology, developed around the 1970's by Christopher Alexander and his colleagues (Alexander 1975).

The schema developed for the Activity Add-on could easily be extended to represent input and output of processes. The Time Settings function of the software also seems useful for *process modelling*. A process modelling function is not implemented in this prototype, since it is a functionality to be used in other contexts than intended here. However it is relevant to mention as a possible extension of the program.

4. ACKNOWLEDGEMENTS

The programming work for the Add-on has been carried out by Balàzs Piri and László Hatvani of Cadprojekt, Budapest, a subsidiary company of the Swedish software developer Lasercad AB. Stefan Larsson and Bengt Larsson of Lasercad have been active in realising this project, which was financed by the Swedish research programme IT Construction and Real Estate Management 2002, together with Lasercad and Lund University. I have had stimulating discussions with my colleague Jonas af Klercker on the subject of user activity modelling in building design.

5. REFERENCES

Akademiska Hus, 2000, "RFP-template", Akademiska Hus, Lund.

Alexander C., 1975, *A Timeless Way of Building*, Oxford University Press, New York.

Bubenko jr J. A., 1993, "Extending the scope of information modelling", Report Nr. DSV 93-034. Department of Computer and Systems Science, KTH, Stockholm.

Bunge M., 1977, *Ontology I: The Furniture of the World*, Vol. 3 of Treatise on Basic Philosophy, Reidel, Dordrecht and Boston.

Bunge M., 1979, *Ontology II: A World of Systems*, Vol. 4 of Treatise on Basic Philosophy Reidel, Dordrecht and Boston.

Bunge M., 1983, *Epistemology and Methodology I: Exploring the World,* Vol. 5 of Treatise on Basic Philosophy, Reidel, Dordrecht and Boston.

Carrara G., Y. E. Kalay and G. Novembri, 1994, "Knowledge based computational support for architectural design", *Automation in Construction*, 3(2-3), p. 157-175.

Checkland P., 1981, *Systems Thinking, Systems Practice*, John Wiley & Sons, Chichester.

Child J., 1984, *Organization: A guide to problems and practice*, Paul Chapman Publ. London.

Eastman C. (1999) *Building Product Models: Computer Environments Supporting Design and Construction*, CRC Pr, London.

Eastman C. M. and A. Siabiris, 1995, "A generic building product model incorporating building type information", *Automation in Construction*, 3(4), p. 283-304.

Flemming U. and S.-F. Chien, 1995, "Schematic lay-out design in SEED environment" *Journal of Architectural Engineering*, 1(4), p. 162-169.

Ekholm A., 1994, "A systemic approach to building modelling – analysis of some object-oriented building product models", in: Björk, B.-C. (ed.) *CIB W78 Workshop, Aug. 22-24 1994, Esbo, Finland.*

Ekholm A., 1987, *Systemet Människa-Byggnadsverk. Ett ontologiskt perspektiv*, Statens råd för byggnadsforskning R22:1987, Stockholm.

Ekholm A. and S. Fridqvist, 2000, "A concept of space for building classification, product modelling, and design", *Automation in Construction,* 9(3), p. 315-328.

Ekholm A. and S. Fridqvist, 1998, "A dynamic information system for design applied to the construction context", in: Björk B.-C. and A. Jägbeck (eds) *The Life-Cycle of IT Innovations: Proceedings of the CIB W78 Conference, June 3-5, 1998.* Royal Institute of Technology, Stockholm.

Ekholm A. and S. Fridqvist, 1996, "Modelling of user organisations, buildings and spaces for the design process", in Turk Z. (ed.) *Construction on the Information Highway. Proceedings from the CIB W78 Workshop, 10-12 June 1996, Bled, Slovenia.* CIB, ?

Habraken J., J. Boekholt, A. Thyssen and P. Dinjens, 1974, *Variations, The Systematic Design of Supports*, MIT Press, Cambridge.

Hales H. L ., 1984, *Computer-aided facilities planning.* Vol. 9 of Industrial engineering, Marcel Dekker Inc., New York.

Hendricx A., 2000, *A core object model for architectural design.* Departement Architectuur, Katholieke Universiteit Leuven, Leuven.

Kalay Y. and C.H. Séquin, 1998, "A Suite of Prototype CAD Tools to Support Early Phases of Architectural Design", *Automation in Construction*, 7(6):449-464.

Kumlin R. R., 1995, *Architectural Programming.* McGraw-Hill, Inc., New York.

Muther R., 1973, *Systematic Layout Planning*, Cahners Books, Boston.

Schenck D. A., and P. R. Wilson, 1994, *Information modelling: The EXPRESS Way*, Oxford University Press, Oxford.

Svensson K., H .Yngve, and C. Bergenudd, 1999, *Förvaltningshandlingar 2000. Slutrapport.* Byggstandardiseringen, Stockholm.

On the road to standardization

Rudi Stouffs and Ramesh Krishnamurti[1]
Delft University of Technology
[1]Carnegie Mellon University

Key words: Information exchange, Standardization, Representations

Abstract: This paper offers an analysis of current standardization efforts, including a classification of their approaches and an evaluation of their advantages and disadvantages with respect to different contexts. In focusing on the design context, a *syntactic* approach to standardization is recommended, and exemplified with a concept for representational flexibility termed *sorts*.

1. INTRODUCTION

Effective digital representations for design have been a topic of research since Sutherland's Sketchpad (Sutherland, 1963) marked the beginning of CAD research. Early efforts into purely geometric representations led to the establishment of geometric modeling as a research field, presenting us, amongst others, with polygon-based and NURBS-based three-dimensional representations that currently form the basis of most modeling applications. More recently, product modeling research has taken a much wider view of design representations, considering geometric design as only one aspect in the product design process and focusing on design as a collaborative process between a variety of actors and experts from many different design disciplines. These different disciplines are concerned with different aspects of the final product and require different representations to work with. Furthermore, different actors adopt different design techniques and methodologies, demanding alternative design representations for the same product aspect. Integrating these different design views into a single product model, or supporting information exchange between alternative representations, possibly in coordination with a central product model, is far

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 75–88.
© 2001 *Kluwer Academic Publishers.*

from straightforward, as current research into product models, such as ISO STEP (ISO, 1994), illustrates.

In architectural and building design, this problem is even more prominent, as design methodologies are varied and diverse, the actors in a collaborative building project are numerous and from a large body of disciplines, and not least, both the project and team are potentially unique from project to project. Furthermore, the building industry is fragmented and characterized by a large number of small- and medium-sized companies, making it even harder to impose common models or processes for information exchange. As a result, information exchange in the building industry has long been, and still is, dominated by a data exchange format, DXF, that is mostly concerned with geometric information and which was designed for use with a single commercial application, AutoCAD™. At the same time, many efforts exist and have existed to conceive a common product model for building design, for example, within the STEP developments, by the International Alliance for Interoperability (Bazjanac, 1998), and more recently in XML (Tolman and Böhms, 2000, aecXML, 1999). Despite the many efforts, little real progress has been made, both in agreeing on common models for various building design aspects, and in convincing the building industry to adopt such models on a general level.

This paper offers an analysis of current standardization efforts, including a classification of their approaches and an evaluation of their advantages and disadvantages with respect to different contexts. In focusing on the design context, a *syntactic* approach (e.g., O'Brien, 2000, Stouffs and Krishnamurti, 1997) to standardization is recommended, and exemplified with a concept for representational flexibility termed *sorts*.

1.1 Data exchange

Between and within disciplines, building partners use a variety of different applications and tools based on many distinct data formats. This diversity of data formats makes supporting information and data sharing within a building project a complex and difficult task. Various approaches to facilitate data exchange among partners exist, based on a number of different techniques and technologies. The most obvious approach is to develop a specific utility for translating data between two given formats. Despite attempts at developing alternative approaches, this is still the most widely used. The advantages are clear: the single purpose supports a focused development towards a highly effective and efficient tool that emphasizes the nature of either or both formats or the specifics of the context in which the utility will perform its task. Such a utility may be used stand-alone or integrated into an application that uses either data format, e.g., in the form of

an import or export functionality, or into a system that offers multiple translation facilities.

Most often, such specific utilities serve data sharing in conjunction with a standard or pseudo-standard. Consider DXF, a data exchange format developed for the purpose of AutoCAD™'s own translation needs between subsequent versions of the software. This format was adopted by the CAD software industry as a pseudo-standard for CAD data exchange. By integrating import and export functionalities from and to DXF into every CAD application, the format serves as a standard for data sharing among CAD applications. The most obvious advantage of such a standard is the fact that each application needs to support translation only between a single pair of data formats, that is, from the proprietary format to the standard and back. However, pseudo-standards are ill suited to this task. As these were never developed for this task, they neither reflect the nature of the proprietary format nor the context of the exchange. For example, DXF supports neither NURBS, a popular geometric representation for curved surfaces, nor textures, making it ill suited for sharing advanced 3D modeling data.

General standards for exchanging building data may overcome these limitations. However, standards are difficult to develop, as these require a broad consensus among industry members. Particularly in the building industry, such consensus is hard to achieve. Many reasons can be thought of. Most commonly, the fragmented nature of the building industry and the uniqueness of each building project (Buckley, Zarli, et al., 1998) are mentioned as primary reasons for this failure to achieve a standard for data sharing among project partners. Equally, neither has hope yet faded. New approaches based on advances in software technology have resulted in renewed and increased efforts and in better chances of achieving such a standard. Object technologies (e.g., Bazjanac, 1998, van Nederveen, 2000) and XML (e.g., Tolman and Böhms, 2000, aecXML, 1999) have served as the catalysts for these activities.

2. STANDARDIZATION APPROACHES

2.1 A-priori versus a-posteriori

Different approaches can be distinguished in standardization efforts. Generally, these adopt an *a-priori* approach: an attempt is made, before or at the onset of the project, at establishing an agreement on the concepts and their relationships which offer a complete and uniform description of the project data. If this collection of concepts and relationships is conceived of independent of the project specifics, i.e., its context, then the approach can

be additionally denoted as *top-down*. The STEP effort is a prime example of a *top-down*, *a-priori* approach, offering a methodology for developing product models and for the exchange of these product models, including its application to various industry domains.

The alternative is a *bottom-up* approach, where the project participants attempt to establish such a conceptualization from practice, based on the project specifics, at the onset of the project. *Object trees* (van Nederveen, 2000) are an example of an *a-priori*, *bottom-up* approach. Primarily aimed at the construction planning phase, object trees serve to improve electronic communication between participants of different disciplines in large-scale construction projects by offering them a methodology for developing representational object trees corresponding to concept hierarchies of construction aspects and elements, and their attributes. The methodology requires all participants to concur on the concepts and attributes involved; in return, it presents them with a unified framework for relating activities and for data exchange among participants. It is specifically suited for the construction and construction planning phases of large-scale projects in which the advantages of the conceptual and representational framework far outweigh the disadvantages of the need for an *a-priori* consensus.

Focusing on the design phase, it is debatable whether an *a-priori* approach, even if successful in the future, will support the variety and flexibility it intends to enable. Conceivably, it may further restrict creativity and individuality by imposing a common product model that caters only to an *a-priori* defined collection of views. For one, new design and analysis techniques or methodologies may be conceived and developed requiring new and different design representations that lie outside of the scope of the product model. Secondly, diversity in design approaches within the same discipline may not be, as a whole, supported by the same model. Lastly, even within the same design process, a single actor may choose to adopt different design representations for different purposes at various stages of this process.

Thus, there is a need to offer both flexibility in representations that allows a designer to adapt a representation to her intentions and needs, and representational dynamism that enables representations to be reconfigured throughout the design process in order to reflect the task at hand. This calls for an *a-posteriori* approach where users are empowered to define their own representations within their design activities and are provided with the tools to, subsequently, communicate the corresponding data into alternative representations as adopted by the other project participants.

2.2 Semantic versus syntactic

In the process of establishing a common product model, two steps – *semantic* and *syntactic* – can be considered. The former refers to the conceptual development, the latter to the translation of this conceptualization into a representational structure for practical use. Standardization efforts tend to focus on one of these steps. For example, the LexiCon effort was primarily concerned with a formal vocabulary for the storage and exchange of information in the construction industry, although representational development is now under way (Woestenenk, 1998, 2000).

S*emantic* development in any standardization effort may serve as the starting point for different *syntactic* developments. For instance, the STEP effort includes the specification of representational structures; other standardization efforts consider adopting the STEP semantics, or even part of the technology, in order to offer alternative syntactical expressions. An international consortium has been founded within the building industry that aims to define an object-oriented data model as a basis for project information sharing in the industry. These efforts of the named International Alliance for Interoperability (IAI) have resulted in a specification of Industry Foundation Classes (IFCs) defining a building object model shared by all IFC-compliant applications (Bazjanac, 1998). The IFC product model was developed using the EXPRESS modeling language developed in the STEP effort, and shares many concepts with the STEP product model developed for the building industry. Recently, the IAI also supports the aecXML Working Group, which started working on an extension of XML, a universal format for structured documents and data for the Web, in order to facilitate data exchange in the building industry (aecXML, 1999). The E-Construct project is also concerned with the development of an XML extension, named bcXML, to support e-commerce in the building industry (Tolman and Böhms, 2000). It builds upon the LexiCon semantics defined for projects and adopts the LexiCon tool for the purpose of a user interface to the data structures.

XML can be considered as an alternative to object technology in order to develop representational structures corresponding to conceptual product models. However, XML is more than a technology; it is a meta-language that serves to define markup languages for specific purposes. By specifying a grammatical structure of markup tags and their composition, a markup language is defined that can be shared with others. When project partners can agree on the tags, they can exchange data described in any markup language based on these tags, even if their own markup language differs in scope or composition. As such, XML may be considered as a *syntactic* standard (O'Brien, 2000). XML can also be considered as an alternative

modeling language to the EXPRESS technology of STEP. XML has the advantage that it is readable both by humans and by the computer. Markup languages based on XML can easily be adapted or extended to one's own specific purposes or needs. Thus, XML structures can easily be defined corresponding to a conceptual product model and such structures can be compared between different models. In fact, it may be quite ironic to consider product model standards as structures fixed in XML (O'Brien, 2000).

3. REPRESENTATIONAL FLEXIBILITY

While these standardization efforts have more or less the same aim, their strategies are quite different each with distinct advantages and disadvantages in supporting flexibility in data formats for varying purposes and needs. The necessity for the support of data exchange reflects a desire to use alternative design representations that enable a particular expression, analysis, or organization. Translation utilities can support data exchange between the standard and proprietary data formats. However, there is a limit to what can effectively be catered for in this fashion. Advances in methodologies, techniques, and technologies repeatedly require new representations of the same building component or building aspect. Standards, however, are necessarily based on current knowledge, uses, and needs. The difficulty in establishing a standard and having it adopted as a basis for data sharing among all or most software applications on the market almost inhibits any subsequent changes in order to update it to new requirements – unless, such flexibility is built into the technology.

The IFC effort attempts to overcome this difficulty by adopting an object-oriented approach and envisioning an evolving object model. Objects encompass both data and access to this data, possibly including operations on the data. Applications can use this model, or parts of it, to define the underlying representation, or incorporate a translation from and to this model into their functionality. When the model is altered through a modification or extension of the object functionality or the development of new object classes, a corresponding adaptation of the applications may not be necessary, unless one wants to make use of the additional functionality provided in the model. In this manner, a single model can respond to advances in knowledge and technology. At the same time, however, this model still depends on a consensus and, as such, will not be able to support the entire spectrum of alternative design representations that can suit particular users or specific situations in the building process. Furthermore, access to this model is only available to software developers and, as a result, a designer will, in most

cases, be restricted to those representations that are provided by the software applications on the market rather than be able to exploit the potential of a truly flexible standard.

The Lexicon model suggests an alternative approach. Though as part of a semantic model, it considers a semi-syntactic approach in which concepts are unambiguously defined by their constituent attributes (Woestenenk, 1998). These attributes then comprise the primitive concepts that define the semantic vocabulary of this model. Taking this descriptive approach one step further, the attributes themselves can be described syntactically, leading to a purely syntactic description of the concepts as compositions of primitive data types. Within a formal structure, these syntactical descriptions may be compared independently of their conceptual meanings, thus allowing for synonym concepts. XML offers such a formal framework. As such, XML allows for an *a-posteriori* and *syntactic* standardization approach, providing all participants with the ability to define or adopt their own data model in XML, and considering ways of translating these different models between one another at a later stage, using tools developed for this purpose.

3.1 A framework for representational flexibility

XML is particularly suited to structure otherwise unstructured information, such as textual data, and to organize information available over the Web. However, it does not provide any information on how to manipulate the data and, as such, is ill suited to represent detailed graphical or geometrical data. Instead, a framework for supporting representational flexibility may be conceived of by borrowing from the different approaches in order to combine their respective advantages. From XML, it may inherit a foundation consisting of an extensible vocabulary of data components that can be composed hierarchically into a representational language. From the IFC effort, it may borrow the object-oriented approach, defining the data components as objects that encapsulate both the data structure and the operations defined on these structures. The symbiosis of these two approaches requires that the compositional operators be defined so that any compositional structure offers the same functionality as each component object separately. Hereto, a behavior can be defined for every component and structure as a collection of common operations on these structures for creation or deletion, or the merging of structures under some formal operations. Through a careful definition of the compositional operators, structures may derive their behavior from their components in accordance to the compositional relationship.

Similar to the IFC approach, a language specification can be derived on two levels. A first syntactic level specifies the vocabulary of primitive object

classes and their respective behaviors. This behavior, in itself, does not provide any meaning to the object class. In fact, a same data structure may define two or more object classes if as many different behaviors can be said to apply, for different purposes. On a second level, a selection of object classes is defined and, individually, named in order to express a semantic concept. These named classes can, subsequently, be composed into a hierarchical structure in order to define an appropriate representational schema. In contrast to the IFC approach, this semantic concept can be specified by the user and the representational structure composed accordingly. Alternative representations can be defined by altering the compositional structure or the selection of component classes. As each representation defines the same common operations, these can be reasonably plugged into an applicative interface for manipulation.

Comparing different representations requires a comparison of the component classes and of the overall compositional structures. At the same time, the expressive power of a representational framework is defined by its vocabularies of primitive object classes and compositional relationships. By carefully selecting the vocabulary of compositional relationships, users can be given the necessary freedom and flexibility to develop or adopt representations that serve their intentions and needs. At the same time, these can be formally compared with respect to scope and coverage in order to support information exchange. Such a comparison will not only yield a possible mapping, but also uncover potential data loss when moving data from less restrictive to more restrictive representations. Translation services can be provided based on both semantic identity and syntactic similarity.

4. SORTS

We are developing such a framework for representational flexibility, named *sorts*. Conceptually, a sort may define a set of similar data entities, e.g., a class of objects or the set of tuples solving a system of equations (Stouffs and Krishnamurti, 1998). Representationally, elementary data types define primitive sorts. These combine to composite sorts under formal compositional operations (Stouffs and Krishnamurti, 1997). The operation of sum allows for disjunctively co-ordinate compositions of sorts, where each sort may be – though not necessarily – represented in the data form; an attribute relationship provides for (recursively) subordinate compositions of sorts in both one-to-many and one-to-one instantiations. Other compositional operations can also be considered, such as an array- or grid-like composition of sorts. The result is a constructive, hierarchical description of sorts as compositions of other sorts, where each leaf node specifies a primitive data

type and every other node defines a compositional operation on its operand children nodes (figure 1).

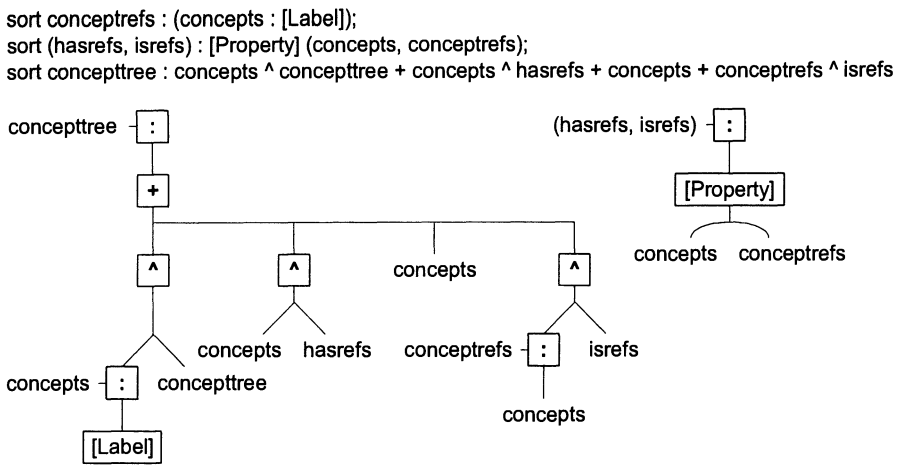


Figure 1. Textual and graphical definition of a recursive concepttree sort. A concepttree may include multiple instances of a single concept, with one instance defined and referenced by all other instances. '+' and '^' denote the operations of sum and attribute, respectively. ':' denotes the naming of a sort. 'Label' and 'Property' are primitive sorts; the latter defines a property relationship sort between two given sorts.

The definition of a sort includes a specification of the operational behavior of its members and collections, denoted as *forms*. The behavioral specification enables a uniform handling of forms of different sorts, on the proviso that the universe of all forms of a sort is closed under the respective operations. Primitive sorts have their behaviors assigned in order to achieve a desired effect, e.g., discrete behaviors for points and labels, an interval behavior for line segments, and an ordinal behavior for weights such as thickness or tones. On the other hand, a composite sort receives its behavior from its component sorts, based on its compositional relationships (Stouffs and Krishnamurti, 1997). The formal relationships between sorts enable the comparison and mapping of sorts as representational structures; the behavioral specification of sorts supports the mapping of information structures onto different sorts, such that the resulting information structures conform to the definition of the respective sorts or representations.

The concept of sorts only specifies a common syntax, allowing for different vocabularies and languages to be created, and providing the means to develop translation facilities between these. For example, a point may be specified with any number of coordinates depending on its dimensionality, its coordinates may constitute integers, reals or rationals, these may be bounded in space, etc. Sorts can be defined accordingly and, based on their

compositional structures, compared and related. For example, the operation of sum specifies a subsumption relationship on sorts, where one sort may match a part of another sort, under sum (Stouffs and Krishnamurti, 1997). Compositional structures under the attribute relationship, if not equal, may be fully (or partially) convertible: the attribute relationship is associative though not commutative. Based on the result of this comparison, translation support can be provided for and data loss monitored. For example, partial conversions always result in data loss; complete conversions may result in data loss depending on the behavioral categories of the constituent sorts.

Alternative design representations can be defined as variations on a given sort, by altering the components or the composition. As an example, consider a representation for a collection of drawings given a sort that defines a single drawing. By specifying an attribute composition with a sort of labels, a named collection of drawings is enabled similar to a set of layers in a CAD application. Alternatively, by specifying an attribute composition with a sort of points or rectangles, a layout can be represented for these drawings (figure 2). One step further, this sort can be modified to enable drawings to relate to parts within other drawings, allowing for detailing relationships to be specified in this layout.

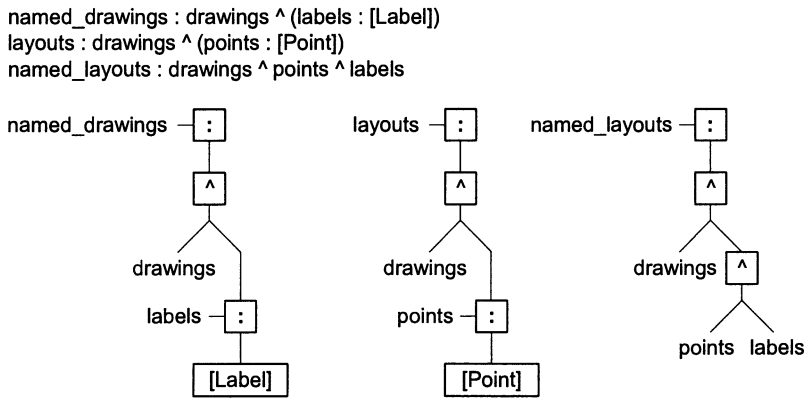


Figure 2. Sort definitions for named drawings, layouts of drawings, and named layouts of drawings, given a sort for a drawing.

As such, there is no imposition of concepts beyond the purely syntactical, and the alphabet of building blocks can be readily extended at all times. No language thus created ever needs to be static. Firstly, a vocabulary may be extended from the existing alphabet or by using newly developed building blocks. Secondly, representations may be updated by reconfiguring the existing composition of sorts or by extending it using additional component sorts. Far from having to redevelop the data structure and the applicative

operations, the concept of sorts aims to provide almost continuous support to evolving representations, providing for an environment that supports exploration and trial, even with respect to the representation. Representational structures can be compared and mapped, data can be readily converted to new and extended (or condensed) representations, and procedural operations remain applicative if such flexibility has been considered.

4.1 Example

Consider design information in the form of design constraints and related information, e.g., for a steel-framed building project (figure 3). This information may be stored in a database organized by type, i.e., constraints, variables, authors, constraint solvers, and other data entities (e.g., images, drawings or explanatory texts), with entities linked as appropriate (figure 4a). This presents an organizationally clean and efficient way of storing design information into a relational database. However, this organization is ill adapted to practical uses. While a representational organization may be dictated by efficiency in data retrieval and management, an effective visualization of the same data depends on user preferences and the task at hand. More important than the distinction between an efficient representation and an effective visualization, is the understanding that different partners in a collaborative environment adopt different views, specify different preferences and use different techniques, while visualizing and manipulating essentially the same information. In the example of the steel-framed building project, the set of design constraints is the result of a collaboration between architect, structural engineer and contractor, to name just a few.

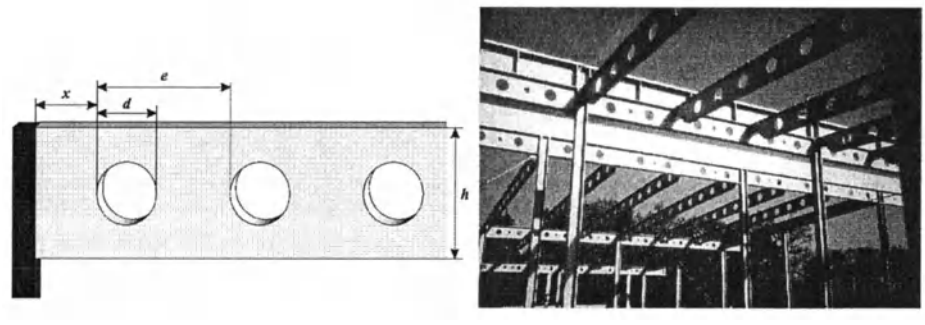


Figure 3. Design problem from a building project: the dimensioning of holes in steel beams.

An alternative visualization of the same project information may take into account the origins of these information entities, that is, for each author

the author's constraints, the constraint solver used, and other data entities provided by this author are specified (figure 4b). Each constraint specifies the variables that are affected by this constraint, and these variables, in turn, link back to the constraints that are defined over these, effectively linking constraints from different authors. Other links, e.g., between constraints and other data entities, can also be maintained and presented.

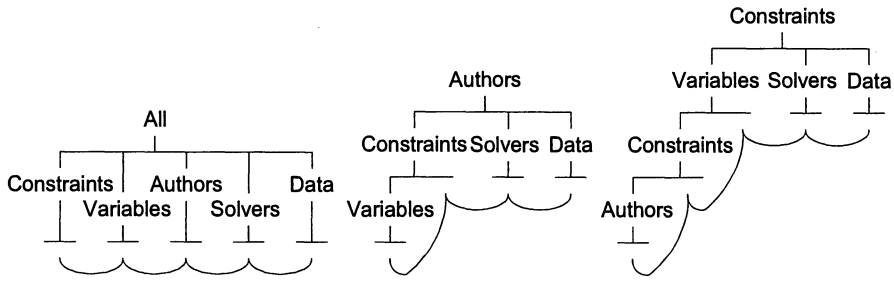


Figure 4. Three different organizational schemes for the same project information: a) by type, b) by author, and c) by design constraint.

Such an organization provides the user with an overview over the different authors' (or domains') contributions. One step further, the effective support of an actual design session may require the design itself, i.e., the design constraints, to form the centerpiece of the visual environment. Other information can be linked from the appropriate constraints in order to clarify each constraint's context and role in the design. A corresponding representation places the author's constraints at the top level (figure 4c). Every constraint specifies the variables affected, the author's constraint solver, and the data related to this constraint. Each variable, in turn, specifies the constraints from other authors that are defined over this variable, and each of these constraints specifies its author. Other links between information entities are additionally provided. This representation allows the author to directly access information related to each constraint. It also enables the user to evaluate the effect of altering a constraint on the design and whether such a change may interfere with other constraints specified by the partners in the collaboration.

As the example attempts to illustrate, *sorts* enable the development of different design views from a same data structure for different users and purposes. In the context of Web presentation, *sorts* can be adopted to prepare the retrieved information appropriately for presentation. Links and connections between information entities are treated as attributes to either or both entities. This approach allows for a uniform and flexible method of presenting information. Figure 5 shows snapshots from a VRML visualization for the steel-framed building project.

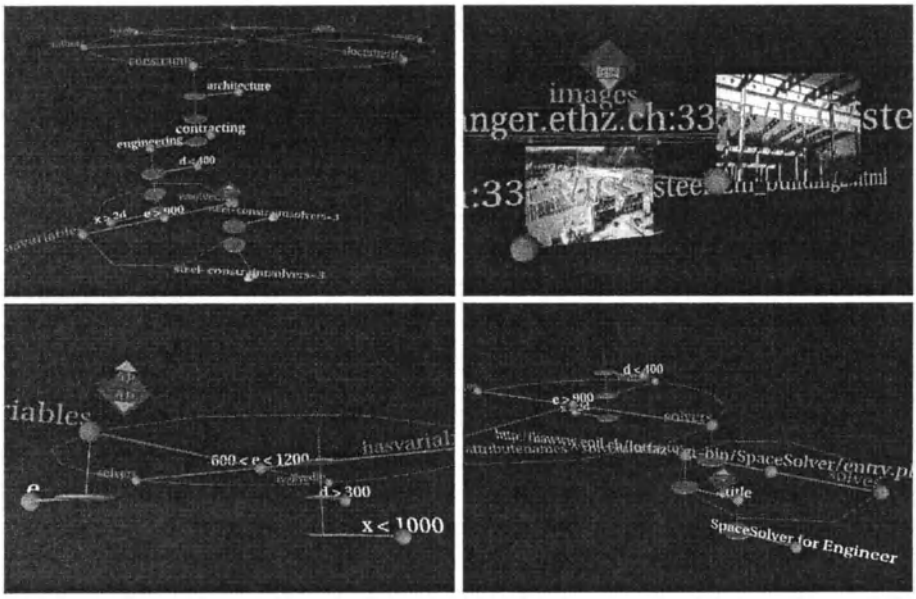


Figure 5. Snapshots of a VRML visualization for the steel-framed building project according to the first and last schemes of figure 4, respectively, a) a set of design constraints and b) related image entities, and c) views of the architect's and d) engineer's constraints. All VRML presentations are generated using an implementation of sorts.

5. CONCLUSION

New technologies, i.e., object technologies and XML, are fueling new interest in standardizing product models for the building industry. However, these same technologies, together with the Internet, reflect a strive for flexibility that stands in contrast to the concept of an all-encompassing standard. XML offers an example of how data exchange can be supported independently of the product models that are applied. *Sorts* attempts to achieve the same flexibility but with increased support for geometrical data and for the comparison of sorts and the translation of data between different sorts.

The concept of *sorts* aims to provide almost continuous support to evolving representations, providing for an environment that supports exploration and trial, even with respect to the representation. By specifying only a common syntax, it allows for different vocabularies and languages to be created, and provides the means to develop translation facilities between these. There is no imposition of concepts beyond the purely syntactical, and the alphabet of building blocks can be readily extended at all times.

6. ACKNOWLEDGMENTS

The research on sorts is partly funded by the Netherlands Organization for Scientific Research (NWO), grant nr. 016.007.007. The example from the steel-framed building project was part of a project funded by the Swiss National Science Foundation, grant nr. 5003-045357, while the first author was working at the Chair for Architecture and CAAD at ETH Zurich. This project constituted a collaboration between ETH Zurich, EPF Lausanne, and various partners from the Swiss building industry.

7. REFERENCES

aecXML, 1999, AecXML: *A framework for electronic communications for the AEC industries*, IAI aecXML Domain Committee. http://www.aecxml.org/technical/

Bazjanac, V., 1998, "Industry Foundation Classes: Bringing software interoperability to the building industry", *The Construction Specifier*, 6/98, p. 47-54.

Buckley, E., A. Zarli, C. Reynolds, and O. Richaud, 1998, "Business objects in construct IT", in: R. Amor (ed.) *Product and Process Modelling in the Building Industry*, Building Research Establishment, Watford, England, p. 117-130.

ISO, 1994, ISO 10303-1, *Overview and fundamental principles*, International Standardization Organization, Geneva, Switzerland.

O'Brien, M.J. and N. Al-Biqami, 2000, "XML, flexibility and systems integration", in: G. Gudnason (ed.) *Construction Information Technology 2000*, Vol. 2, Icelandic Building Research Institute, Reykjavik, Iceland, p. 656-661.

Stouffs, R. and R. Krishnamurti, 1998, "An algebraic approach to comparing representations", in: J. Barallo (ed.) *Mathematics & Design 98*, The University of the Basque Country, San Sebastian, Spain, p. 105-114.

Stouffs, R. and R. Krishnamurti, 1997, "Sorts: a concept for representational flexibility", in: R. Junge (ed.) *CAAD Futures 1997*, Kluwer Academic, Dordrecht, The Netherlands, p. 553-564.

Stouffs, R., R. Krishnamurti, and C.M. Eastman, 1996, "A formal structure for nonequivalent solid representations", in: S. Finger, M. Mäntylä and T. Tomiyama (eds.) Proceedings of *IFIP WG 5.2 Workshop on Knowledge Intensive CAD II*, International Federation for Information Processing, Working Group 5.2, p. 269-289.

Tolman, F.P. and H.M. Böhms, 2000, "Electronic business in the building-construction industry: preparing or the new Internet", in: G. Gudnason (ed.) *Construction Information Technology 2000*, Vol. 2, Icelandic Building Research Institute, Reykjavik, Iceland, p. 928-936.

van Nederveen, G.A., 2000, *Object trees: improving electronic communication between participants of different disciplines in large-scale construction projects*, Delft University of Technology, Delft, The Netherlands.

Woestenenk, K., 1998, "A common construction vocabulary", in: R. Amor (ed.) *Product and Process Modelling in the Building Industry*, Building Research Establishment, Watford, England, p. 561-568.

Woestenenk, K., 2000, "Implementing the LexiCon for practical use", in: G. Gudnason (ed.) *Construction Information Technology 2000*, Vol. 2, Icelandic Building Research Institute, Reykjavik, Iceland, p. 1049-1057.

Dynamic Retrieval in an Urban Contextual Databank System using Java-CGI Communications

Development of the SUCoD Prototype

Chengzhi Peng, David C. Chang, Peter Blundell Jones and Bryan Lawson
University of Sheffield

Keywords: Urban Contextual Databank, Dynamic Retrieval, Java, Common Gateway Interface, VRML, HTML, Virtual Cities

Abstract: This paper presents our current development of the Sheffield Urban Contextual Databank (SUCoD) prototype that provides users with a Web-based interface for dynamic retrieval of architectural and urban contextual information of the city of Sheffield. In comparison with past attempts of building Virtual Cities accessible over the Internet, we have experimented with a different system architecture capable of generating VRML models and other related documents *on the fly* according to users' requests. Through working examples, we describe our methods of implementing the data communications between Java applets and Common Gateway Interface (CGI) scripts. The SUCoD prototype has been developed to explore and demonstrate how *user centred dynamic retrieval of urban contextual information* can be supported, which we consider a user requirement of primary importance in its future use for collaborative design and research relating to the city of Sheffield.

1. INTRODUCTION

In the past three years, the Diploma students at the School of Architecture at the University of Sheffield have embarked on the Sheffield Urban Study project, researching and reconstructing parts of the City of Sheffield at around the turn of the twentieth century. In each year's program, the students were organised to work in groups to produce research reports, drawings and scaled physical models of the buildings, streets and places by investigating all sorts of local historical literature and archives. Apart from creating a substantial research database on the Sheffield urban history, the

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 89–102.
© 2001 *Kluwer Academic Publishers.*

students also learned how to undertake historical research in a collaborative and systematic manner (Blundell Jones, Williams, et al., 1999).

To date, the database resulting from the urban study project covers a region of 40 squares (one square of 200m each side), reaching 160 hectares in total. Subsequently, a public exhibition of the project work has been held at the Mappin Art Gallery in Sheffield for several months. In June 1999, a major research grant from the Arts and Humanities Research Board was awarded to the authors to embark on a separate project with an aim to investigate how the physical database amassed by the Sheffield Urban Study project can be put into electronic form accessible through multiple routes. The project was set out to achieve the following objectives:

- To further evaluate and elaborate the hypermedia-based modelling and authoring methodology established in the previous pilot study (Peng and Blundell Jones, 1999).
- To deliver a prototype of an interactive hypermedia databank that will provide user-friendly facilities for searching and retrieving digital documents created on the basis of the Sheffield Urban Study Project.
- To make the databank accessible nationally and internationally via the Arts and Humanities Data Service and the Internet.
- To demonstrate that a dedicated urban contextual information system of a region can facilitate collaborative research and design regarding the historical or contemporary urban development of that region.

Our pursuit of the research has led to the development of the first prototype named as "Sheffield Urban Contextual Databank" (SUCoD). The SUCoD prototype presents features that are similar to some of the projects related to building "Virtual City" pursued by other research groups. In building the Sheffield model, however, we are particularly interested in developing novel spatial-oriented approaches to retrieving *dynamically* the urban contextual information resulted from the previous research into the city's history. By adopting the Java™ and Common Gateway Interface (CGI) technologies, we have implemented in SUCoD a Web-based computing framework on which a large amount of contextual datasets can be deposited and maintained constantly. The prototype has been tested to demonstrate the feasibility for users to dynamically retrieve contextual information of any areas of the city through an interactive map interface. Users can browse and navigate city contexts as *dynamic Web contents* that display historical maps, virtual reality models, and other digital resources related to the specific buildings or areas retrieved by the users.

In this paper, we shall present and discuss our current design and implementation of the SUCoD prototype, focusing on our development of its functionality and user interface. The remainder of the paper is organised as follows: In Section 2, a brief review of three related projects on building virtual cities is presented, highlighting the various purposes and intended

uses of the virtual cities created. Following a discussion on some of the major system design issues, we describe in Section 3 the stages of developing the SUCoD prototype. Finally, Section 4 reports on the initial responses and feedback from the audiences to whom we have made demonstrations.

2. BUILDING VIRTUAL CITIES: RELATED PROJECTS AND SYSTEM DESIGN ISSUES

In recent years, the constructing of "virtual cities" has become an active research and development topic in architectural computing and urban studies. An important feature of virtual cities is that they are accessible over the Internet and they present users with 3D virtual worlds based on the graphic data format of VRML (Virtual Reality Modelling Language, see [WWW01] for the latest specification of the language). Apart from being Internet-HTTP compatible, VRML provides many other features that modellers can exploit to create rich 3D virtual environments in which the end users have controls over various modes of navigating the models (e.g., walk, fly, pan, rotate, etc).

Applications of virtual cities or virtual environments have been popular in domains where 3-dimensional simulated spaces and architectural metaphors can offer new kinds of graphical user interface. In academic research, on the other hand, the enterprise of building virtual cities seems be driven mainly by its ability to represent/simulate historical or contemporary architectural and urban environments. Past attempts show that constructing virtual urban environments can be an effective way of creating and maintaining a large organisation of online urban and architectural information. Virtual cities can be created to provide users with convenient routes to the models and related resources related to the historical or contemporary aspects of the cities. In the following, we shall briefly look at three related projects.

2.1 Three projects on building virtual cities

The **Bath Model** was created by the CASA group at the University of Bath (Day, Bourdakis, et al., 1996). The aim was to visualise the changes in the City of Bath and how the city was originally constructed. The Bath model was initially built with 3D modelling and texture mapping in AutoCAD, and was subsequently converted into a single VRML model. The model covers a square of 10 km each side with texture-mapped terrain

around the city and the Bath abbey in some detail. The model was organised in four levels of detail with data links to images and texts on the city and building history [WWW02]. The Bath model has been used by the local planning authority for development control and public debate on future city development (Bourdakis and Day 1997).

The **Glasgow Directory** was developed by the ABACUS group at the University of Strathclyde [WWW03]. The system was intended to present the city of Glasgow in 3D form with architectural details and a searchable citywide database of her important architectural heritages. A variety of online resources are provided, including 3D volumetric models (in VRML 2), digital photos of architectural interiors assembled in QuickTimeVR™, a map for browsing and retrieving the VRML models, and a search engine for its database of significant buildings of the city. As on the Web, the Directory now contains a total of 47 separate VRML model segments, covering an area of 25 km^2 accurate in height and plan. The Glasgow Directory has been used by developers' visualising proposals in the context of the city and also to promote the City and its architecture via the Web.

The **Virtual Los Angeles** project undertook by the Urban Simulation Team at UCLA aimed to achieve a high quality community and city visual simulation for the City of Los Angeles [WWW04]. By combining aerial photographs with street level imagery, databases of trees, and 3D geometry, realistic urban neighbourhoods were created. The system provides substitution operations for interactive evaluating alternatives and temporal modelling in simulating existing neighbourhoods, historical reconstructions, or proposed new developments. The ultimate goal is to deliver multiple linked VRML models, covering the entire LA basin (10,240 km^2). The project intends to deliver a real-world application capable of supporting hundreds of remote simultaneous interactive users, working in civil engineering, urban design and planning-related problem solving (Snyder and Jepson 1999). There was also the plan of allying the system with other citywide applications such as emergency response, virtual learning environments, etc.

2.2 System design issues

The quick look above is by no means exhaustive. An extensive review of more virtual city projects deserves a separate paper to examine properly the issues and complexity involved. However, related to our interest in building the Sheffield model, the previous projects suggest a number of system design issues of immediate concerns:

a) *VRML and accessibility over the Internet.* The VRML 2 file format was adopted by all three projects in support of wider user accesses through

the Internet and the Web. Given that special software for displaying and interacting with VRML worlds are now widely available as free Web browser plug-ins, the acceptance of VRML in building and accessing virtual cities seems here to stay for sometime to come. To our knowledge, no alternatives have been proposed to replace the VRML as a non-proprietary file format for representing 3D worlds on the Internet. Rather recently, the Java 3D™ technology has evolved into a full strength programming language for building network-centric and scene graph-based 3D applications. Java 3D, however, does not define a file or network format of its own; it is designed to provide a lower-level underlying platform API, and many VRML implementations are expected to be layered on top of Java 3D [WWW05].

b) *3D models and other data types*. All three projects have shown that a 3D model is not the only type of dataset required in building virtual cities. Other types of data such as maps, photos, textual documents, and video/ audio clips, can be as important, as they provide aspects of urban information that cannot be recorded effectively in 3D modelling.
c) *VRML navigation and information retrieval*. Interlinking the different types of digital resources created is an important issue. A combined use of the hyper linking in HTML and VRML facilitates users' navigation and retrieving information with a spatial context. The Glasgow Directory clearly demonstrates the usefulness of this facility.
d) *Spatial subdivision in urban modelling*. Subdividing a city region into smaller areas seems necessary in managing the modelling process. Building a city region into a single large file poses at least two problems: difficulties in maintaining the model, and the lengthy download time, which might put off users of the models. The Bath model seen above is an example of a single-file approach with no subdivisions used. Both the Glasgow and Los Angeles models imposed spatial subdivisions, resulting in multiple VRML models that users can retrieve separately. However, we did not find explicit reasons given as how and why the subdivisions were introduced in the first place.

In comparison with the city modelling projects seen above, our Sheffield modelling project is intended to support collaborative research and design in relation to the city contexts. This implies that we should provide not only an adequate urban data repository of a good quality but also a flexible way of retrieving the contextual information for any area of the city. After all, what constitutes an urban context that is relevant or irrelevant to the objectives of a design or research project is open to the participants' interpretations and analyses; it is almost certain that different individuals will define and delineate differently what an urban and architectural context actually consists of (for instance, location of focal point, boundary, shape, and

viewpoints, to say a few). It is therefore, in our view, a critical requirement that the Sheffield model should be capable of allowing users to retrieve 3D models and other related contextual resources according to user-defined locations and boundaries of relevance.

Looking further into the requirement of user-centred dynamic retrieval, we realised that while satisfying the need for speedy data transmissions through the Internet, if the separate model sets cannot be combined or re-assembled in ways as requested by the users, the usefulness of these models can be drastically reduced. For example, there will be difficulties if the users' focal points of interest happen to lie on or somewhere near at the lines of subdivision where separate models actually end or meet. We think a better result can be achieved if the users are able to retrieve 3D models together with related information in which the focal points and boundaries requested are located at positions intended by the users.

The points raised above imply that a system should to be capable of generating the models and related information sets "on the fly" according to individual users' retrieval requests. We believe that the urban contextual models and information retrieved in such a manner should better serve the users' needs of contextual studies and modelling. A true test of the usefulness of the online resources that are going to develop lies in whether the contextual information retrieved can serve the users' construction of their own design arguments and narratives. According to our present review, we have not yet seen any virtual cities published on the Web in which users can perform dynamical retrieval of 3D models. In the next section, we shall present how we have achieved this functionality in the Sheffield project.

3. DESIGN AND IMPLEMENTATION OF THE SUCoD PROTOTYPE

Our system development at the early stage was most concerned with how we might build a system that would provide the functionality allowing for dynamic retrieval of 3D models in VRML 2.0. We started off with a simple example on the basis of an HTML form sending messages to a Perl script through the Common Gateway Interface (CGI) protocol. We then progressed to further extended examples by replacing the HTML form with Java applets.

3.1 A simple cylinder-on-board example

Our first working example considered a 3D model of eight simple cylinders sitting on a rectangular board. By selecting particular cylinders, a

user can retrieve VRML models, showing only the cylinders selected together with the board. The task was simulated in an HTML form, which presents an image of the 3D model, eight check boxes for selecting the cylinders, and "*Submit*" "*Reset*" buttons. A CGI script was written and deployed on a server machine, which can be invoked by user's pressing the *Submit* button on the HTML form. On receiving a request sent from a client's HTML form that specified which cylinders were picked (including none), the CGI script was executed to generate a VRML model, containing only the selected cylinders. The VRML model was then transferred and loaded to the user's Web browser when he or she clicked on the hyperlink pointing to the resultant VRML model generated on the server.

Figure 1 is a screen shot of the cylinder-on-board example, showing the HTML form (left) and the retrieval of a VRML model (right) in two separate Internet browser windows. The VRML model was viewed in the Cortona VRML Client 3.0 (by ParallelGraphics [WWW06]).

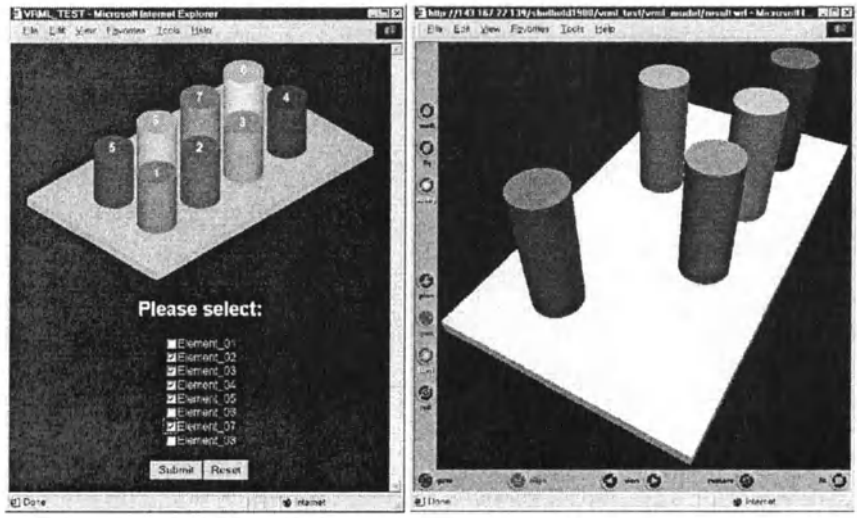


Figure 1. Exploring dynamic retrieval of VRML models in the cylinder-on-board example. On the left is the HTML form for users to select numbered cylinders (0-8) via the eight check boxes. For each selection submitted, a VRML model is generated *on the fly* by a CGI script on the server; the VRML model is then displayed to the user in a separate window (right).

Our exploring the above example appears significant in several aspects:

a) All the VRML models were generated on the fly upon receiving client's requests sent through the HTML form interface. The server did not store any pre-built VRML models (if it did, we would have to pre-build a total of 256 files of static VRML models on the server to be capable of responding to all possible user selections).

b) CGI programming with Perl 5.6 (as in ActivePerl 5.6 provided by ActiveState [WWW07]) worked fine for our objectives of processing user requests and constructing dynamically well-formed VRML files. A VRML file can be generated by a Perl script that matches and combines data (DAT) files in which the basic 3D data of individual elements or objects (i.e., the cylinders and board in our example) are stored.
c) The cylinder-on-board example may look trivial but it did bear some degree of correspondence to our modelling of a city environment. Basically, the cylinders refer to individual buildings, and the board to a city's terrain. Although we did not subdivide the board into smaller blocks, we were sure it could be done just like the cylinders. The entire terrain of an urban area, therefore, can be subdivided into smaller elements and retrieved separately just like individual buildings.

With two machines acting as a server (a Pentium III running the Internet Information Server 2.0 under Windows NT 4.0) and a client connected to the Internet through the Ethernet at the School, the above working example proved our initial idea of dynamic retrieval workable. The generating and display of VRML models across the network has been reliable. But two immediate issues were sensed following the working example:

1. The check-box type of interface will not be usable if the model is extended to include, say, a hundred cylinders. How is a user supposed to deal with one hundred check boxes for selecting or de-selecting particular objects in order to retrieve a VRML model? This issue is important as a city model typically contains thousands of objects if not more. An alternative interface has to be sought to enable user selection on a large sum of individual elements.
2. What will happen to the speed of dynamic generation of VRML models if the amount and complexity of elements far exceed the current working example? To find out more, we have worked on further experiments.

3.2 An extended cylinder-on-board example

Given our purpose of modelling the historical city of Sheffield, we considered it logical to move onto a map-based interface for user selection. With a digital map, a user can simply point and drag a rectangle on the map as a way of making selection. All elements located within the rectangle are interpreted as being selected. There was also the need of basic map-reading operations such as *zoom-in* and *out*, *pan*, *fit all*, etc. In our search for digital map building and viewing tools, we considered a Java™ enabled mapping method worth trying. Basically, a Java-enabled map is built on Java's Applet technology that provides a client-side graphical user interface for displaying and interacting with object-oriented vector-based maps loaded into the

applets. To avoid implementing a Java-based map system from scratch, we chose to work with the ILOG JViews Component Suite [WWW08] as the basis of building our own Java-enabled map applications. Built with ILOG JView's Maps Beans, an applet was first implemented to display a multi-layered interactive map. The applet provides a viewing area for displaying the Java-enabled map composed in ILOG's Composer package. *Figure 2* shows the Java map applet running the previous eight-cylinder example. Several graphical user interface (GUI) components, including the *Control Bar*, *Map Legend*, *Overview* and a *Coordinate Viewer*, were implemented to provide the interactive functions of manipulating the map (for instance, *select*, *zoom in* or *out, fit all* in the *Control Bar*, check or uncheck particular layers of map in the *Map Legend*, and a fast map navigation in the *OverView* panel).

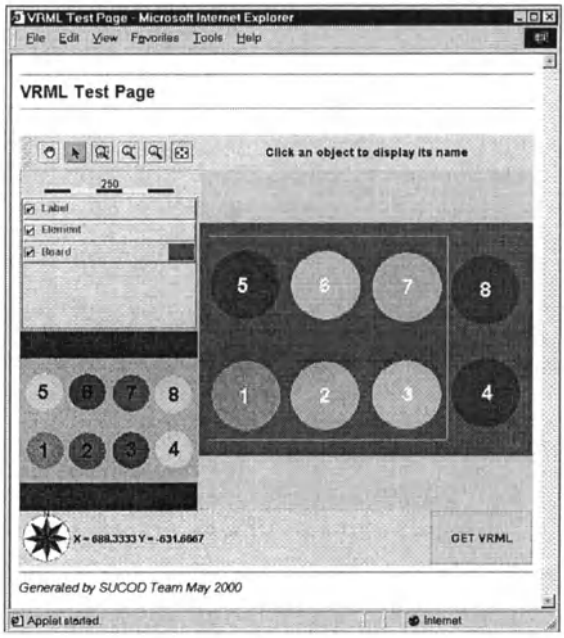


Figure 2. The *VrmlTest* Java applet was developed to experiment with a multi-layered interactive Java-enabled map to communicate with the CGI script. The cylinders can now be selected on the map by the *drag a rectangle and select* method.

However, the idea of a Java-map applet would not work if there were no communications between the map applets and the CGI Perl scripts. As shown before, the HTML form-based communications with the CGI script was essential in dynamic generation of VRML models. A way of establishing the communications between our Java map applets and the CGI

Perl scripts must be found in order that the entire scheme could succeed in achieving our aim. After numerous coding attempts, we finally had the Java code with which the two-way data communications between the Java map applet and the CGI script was established. The "*GET VRML*" button is the GUI component for invoking the Java-CGI communications. With this functionality in place, a user can now make selections on the basis of a map and VRML models will be generated according to what objects are selected within the Java map management environment. *Figure 3* shows our current implementation of the extended cylinder-on-board example. Note that the Java map and the VRML model are now shown in two frames (left and right) on a single Web page.

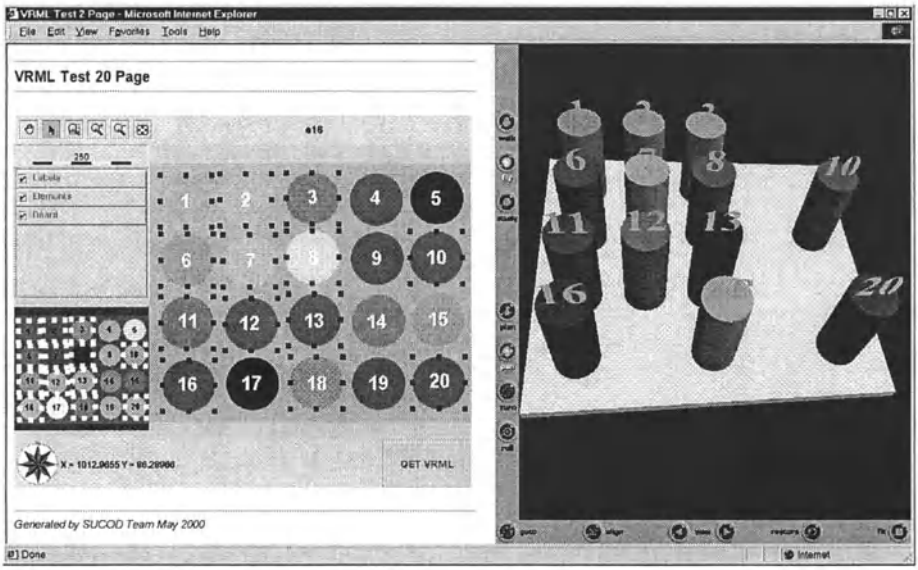


Figure 3. The extended cylinder-on-board example (increasing the number of cylinders to 20). The left frame is for selecting elements on the Java map and the "GET VRML" button is for invoking Java-CGI communications. The right frame is for displaying resultant VRML models.

Again, in the extended example, there were no pre-built VRML models stored on the server awaiting calls from clients. As before, VRML models are generated on the fly by invoking the Perl script hosted on the server. No changes were required in the Perl program when we switched from to the HTML form to the Java map applet. However, it did take considerably longer in downloading the applet as it contains all the Java GUI classes needed for initialising the applet in a Windows environment. In terms of generating and displaying VRML models, the Java-map approach did not

show any significant delay in comparison with the HTML form-based retrieval.

3.3 Work on the SUCoD prototype

Having worked through the two cylinder examples, our next phase of system development was to populate the Java-CGI system framework with real datasets acquired in the Sheffield Urban Study project. Two squares (labelled A2 and B2 originally) were chosen in trying out our first Sheffield Urban Contextual Databank (SUCoD) prototype. The datasets created consists of the following types: (i) the Java interactive map, (ii) data for generating VRML models, and (iii) data for generating HTML pages. *Figure 4* shows a screen shot of the SUCoD prototype when accessed in Microsoft® Internet Explorer 5.

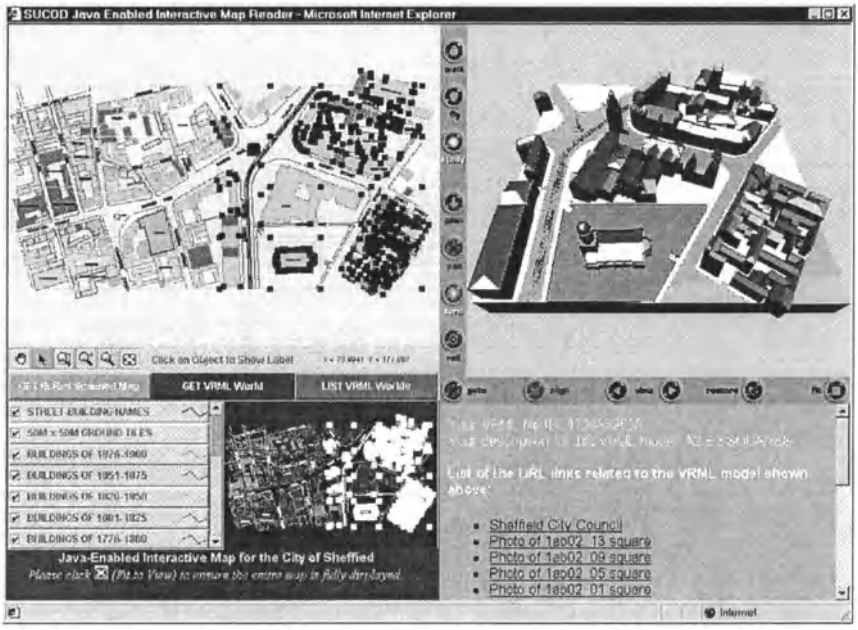


Figure 4. The SUCoD prototype runs in Microsoft Internet Explorer 5. The left frame is the applet for managing the Java map, the right-upper frame for loading VRML models, the right-lower frame for listing links to HTML pages related to the VRML model displayed. On the Java map, "graphic handlers" in tiny black squares are shown for every object selected.

In implementing our first release of the SUCoD prototype, we made the following design decisions on the data types and organisations to be supported by the Java-CGI framework:

a) **The spatial subdivision scheme**. The original spatial subdivision in the Sheffield Urban Study project was a grid of 200m squares across the region being studied. All the physical modelling and paper-based documentation tasks were coordinated in accordance with the grid system. In developing the SUCoD prototype, we decided to adopt a grid of 50m squares so that a smaller granularity for user selection could be achieved. Another reason for adopting the smaller square unit is to coincide with the UK Ordnance Survey Mapping Service which publishes digital urban maps (OS Super Plan®) in 50m by 50m grid [WWW09]. This means, in SUCoD, the smallest unit a user can select is a 50m-square (i.e., a single "ground tile" see below). However, the unit of 50m-square only applies to the subdivision of terrain. Every single building shown on the Java map was treated separately so that a user can pick and un-pick all buildings individually.

b) **The layering of the Java map**. The Java-enabled map was an important aspect of data design, as it should accurately reflect the research conducted in the Sheffield Urban Study project. It should also provide the interface for users to navigate and select areas of interest according to different criteria. By gathering all the data available, we have come up with a Java-enabled map consisting of 20 layers, each of which can be turned on and off separately when shown up in the applet. A total of 7 layers were created to depict buildings according to the various periods of Building Ages (ranging from pre-1750 to 1900) as researched in the Sheffield Urban Study project. Another 8 layers were created to indicate buildings of various Building Usages (Public, Residential, Heavy Industries, etc). A layer named "50m Square Ground Tiles" was created to show the 50m square grid plan as imposed onto the original A2 B2 squares. Like the buildings shown on the various layers of building ages, the "ground tiles" are objects that users can select to retrieve VRML models representing the 3D terrain and streets on it. Finally, the names of all buildings and streets as indicated in the historical maps were added as the top layer to aid user navigation.

c) **VRML models of terrain and buildings**. To carry out sophisticated 3D modelling of the terrain, streets, and buildings, the MicroStation/J™ CAD package was used to generate the basic 3D geometric and colour rendering data. These were then taken as the raw data for a new Perl script to construct a VRML model. Apart from the smallest granularity of selection allowed (i.e., a single building or a single 50m square ground tile as depicted in the interactive Java map), there are no other constraints on what a user can select and retrieve. It is entirely up to the users to decide what buildings or terrain units should be included when retrieving VRML models. From the system's point of view, it is perfectly legal to generate a VRML model with only buildings without the terrain

underneath and visa versa. A user's selection may also result in a VRML world of disconnected terrain units, which do not add up to form a rectangular boundary.

In the course of developing the SUCoD prototype, we are aware that more advanced technologies like Java Servlets™ and JavaServer Pages™ (JSP) have been put forward to replace the CGI approach in implementing a typical Internet-based server-client application (see Hunter and Crawford, 1998; Hall 2000, for example). Our primary objectives at this stage, however, are to explore and evaluate the kinds of system functionality both from the data creation as well as usability points of views and to deliver a data repository holding contextual resources of a high research quality.

4. PROTOTYPE DEMONSTRATIONS AND FEEDBACK

In the past few months, we had the opportunities to make demonstrations of the SUCoD prototype to students, staff and visitors at the School of Architecture. The initial responses from the audience have been largely positive. Following the feedback, the prototype has been continuously revised in response to several points raised during the discussions:

1. *Speed of downloading and reliability.* During the demonstrations, we experienced a lengthy waiting time for initiating the Java map applet. There was also the problem of reliability: from time to time the system seemed not to be responding for reasons not entirely sure to us: networking congestion during the demos, the capacity of the Web Server in use, or the capacity of the client machine in use, etc. In our attempts to ease the problems of speed and system reliability, we decided to migrate SUCoD to the JAVA™ 2 Platform with the latest Java Runtime Environment provided in the JDK™ 1.3.0_01 [WWW10]. With the installation of the Java™ Plug-In 1.3.0_01 [WWW11] on a client site, the change has resulted in a shorter applet initialising time and more reliable performance during interactive sessions.
2. *Relating to contemporary urban context.* Several people have raised the point regarding the contemporary urban context of Sheffield. They suggested that the SUCoD system would not be very useful if it contains no contemporary contextual information. The Sheffield Contextual Databank project was funded to create online resources based on the Sheffield Urban Study project. However, we believe that we have come close to establish a generic system framework in developing the SUCoD prototype. Adding contemporary contextual data should not render the

current framework obsolete but can be treated as an extension to the existing databank. The SUCoD prototype has demonstrated that contextual data acquisition and accumulation can be sustained in a bottom-up manner whether looking into the past, present, or future.

3. *Scanned historical maps*. In studying the Sheffield city history, historical maps have been used extensively to establish the footprints of the historical urban fabric. In constructing the multi-layered interactive Java map and the 3D models, historical maps played a crucial role in laying the foundation. Suggestions were made to us that the historical maps could also be made available through the SUCoD system. We recognised this as an important resource in addition to the VRML and HTML files. Further work on SUCoD is now underway to include a facility for dynamic assembling of scanned historical maps according to user selection.

5. REFERENCES

Blundell Jones, P., A. Williams and J. Lintonbon, 1999, "The Sheffield Urban Study Project", *Architectural Research Quarterly*, 3(3), p. 235-244.

Bourdakis, V. and A. Day 1997, "The VRML model of the city of Bath", in: R. Coyne, M. Ramscar, J. Lee and K. Zreik (eds.) *Design and the Net: Proceedings of the Sixth International EuropIA Conference*, p. 245-259, 2-3 April 1997, Edinburgh, UK.

Day, A., V. Bourdakis, and J. Robson, 1996, "Living with a virtual city", *Architectural Research Quarterly*, 2(1), p. 84-91.

Hall, M., 2000, *Core Servlets and JavaServer Pages (JSP),* Prentice Hall PTR/Sun Microsystems Press: New Jersey.

Hunter, J. and W. Crawford, 1998, *Java Servlet Programming*, O'Reilly UK, London.

Peng, C. and P. Blundell Jones, 1999, "Hypermedia Authoring and Contextual Modeling in Architecture and Urban Design: Collaborative Reconstructing Historical Sheffield", in: Osman Ataman and Julio Bermudez (eds.) *Media and Design Process: Proceedings of ACADIA'99*, p. 116-127, 28-31 October, Salt Lake City, U.S.

Snyder L. M. and B. Jepson, 1999, "Real-Time Visual Simulation as an Interactive Design Tool", in: Osman Ataman and Julio Bermudez (eds.) *Media and Design Process: Proceedings of ACADIA'99*, p. 352-353, 28-31 October, Salt Lake City, USA.

[WWW01] "http://www.vrml.org/VRML2.0/FINAL/" The Virtual Reality Modeling Language Specification, Version 2.0.

[WWW02] "http://www.bath.ac.uk/Centres/CASA/" The Bath Model.

[WWW03] "http://iris.abacus.strath.ac.uk/new/gintro.htm" The Glasgow Directory.

[WWW04] "http://www.ust.ucla.edu/ustweb/projects.html" The Virtual Los Angeles Project.

[WWW05] "http://java.sun.com/products/java-media/3D/" Java 3D™ 1.2 API.

[WWW06] "http://www.parallelgraphics.com/products/cortona/" Cotona VRML Client.

[WWW07] "http://www.activestate.com/Products/ActivePerl/" ActivePerl 5.6 by ActiveState.

[WWW08] "http://www.ilog.com/products/jviews/" ILOG JViews Component Suite.

[WWW09] "http://www.ordsvy.gov.uk/home/index.html" Ordnance Survey.

[WWW10] "http://java.sun.com/j2se/1.3.0_01/index.html" Java™ 2 SDK, Standard Edition (J2SE) Version 1.3.0_01.

[WWW11] "http://java.sun.com/j2se/1.3.0_01/jre/index.html" Java™ 2 Runtime Environment (J2SE) Version 1.3.0_01.

Design through Information Filtering

a search driven approach for developing a layperson's CAAD environment

Sheng-Fen Chien and Shen-Guan Shih
National Taiwan University of Science and Technology

Key words: Information filtering, Design knowledge encapsulation, CAAD for non-designers

Abstract: We propose a CAAD environment for non-designers. It is a new way to enable effective user participation during the design process. This CAAD environment contains an encapsulation of design knowledge and utilises information filtering as an interface to the design knowledge. Two prototypes are implemented as testbeds. So far, our experience has suggested that the approach has a promising future.

1. INTRODUCTION

Architects design by searching through some spaces of designs to find feasible solutions. Clients (usually non-designers) then search through alternative solutions (provided by architects) to identify their desired solutions. In essence, both architects and clients are solving design problems; nevertheless, the corresponding search spaces differ significantly.

Traditionally, the clients' search spaces are fairly small, which often contain five to three, or sometimes only one, solutions. For some design projects (for example, private houses), successive communications between clients and architects will allow architects to put more alternatives into clients' search spaces. Yet for many other design projects (for example, public housing), actual clients (i.e. the eventual users of these buildings) have very little chance communicating with architects and are forced to accept solutions that are not suitable for their needs.

We are interested in strengthening the communication between architects and users so that the resulting designs are tailored to user needs. User

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 103–110.
© 2001 *Kluwer Academic Publishers.*

participatory design seems to be the answer, but it is very costly in terms of time and finance. Besides, to achieve effective communications between designers and non-designers (which users often are) is itself a challenging task. We suggest a new way to look at user participation: by enabling users to make a wider range of design decisions intelligently, that is, providing them a search space with rich design solutions and a mechanism to search through this space for desired solutions.

2. DESIGN KNOWLEDGE ENCAPSULATION

How can "providing users a search space of designs" address the communication issue between designers and non-designers?

Archea (1987) describes the way architects design as follows: "[t]hey seek sets of combinatorial rules that will result in an internally consistent fit between a kit of parts and the effects that are achieved when those parts are assembled in a certain way." In a way, design solutions—those parts and assemblies of parts—have captured certain design knowledge in their existence.

The encapsulated design knowledge is not the general design knowledge at all; rather it is the knowledge for a specific design project. Therefore, for a different design project, an architect should provide a different search space of designs. To construct such a search space, a systematic structure is needed. We find Habraken's (1972, 1998) *open building* concept, in particular the hierarchically refined support-infill relationship, very suitable to form the underlying structure of the search space.

Under this concept, building components (parts) are organized into a hierarchical structure according to the support-infill relationship. "Support" is a fix context (e.g. the structural frame of a building), within which "infill" (e.g. wall panel) can be attached, removed or substituted. "Infill" can become the "support" (e.g. wall panel) of its lower-level "infill" (e.g. window). Organizing design knowledge in such a hierarchical fashion accommodates the knowledge of structural design as well (Gomez, 1998).

3. INFORMATION FILTERING

What technology is suitable as an interface to the encapsulated design knowledge?

The development of *information filtering* has grown from techniques to dynamically display information relevant to a user's interest, and become an essential information technology to support knowledge management (Borghoff and Pareschi, 1998). The design knowledge encapsulated in a search space has to be presented in a way that is understood by non-designers. In addition, we believe that the complexity of design knowledge should be hidden. This means that users should see the system (the interface) as a glass box, within which the lower level component (the search space) acts as black box (Karlgren, Hook, 1994).

Given the hierarchical structure of encapsulated design knowledge, the information filtering process will begin by locating the "supports" of a particular state. From this state, users can explore variations of "infills". By interactively identify filtering criteria, the system may lead a user to specific design solutions.

4. WIDE

WIDE, a web-based interactive design environment, is designed based on the "search" behavior of designers and non-designers to support user participation in the development of apartment buildings (Chien and Shih, 2000). Most apartment units in Taiwan are sold before ever been built. Apartment buyers can customize their units until the construction takes place. This customization process has become a very unique form of user participation. WIDE is intended to be a CAAD environment for non-designers. By working closely with several building developers, we have analyzed the activities and information flows in the customization process (Shih and Chien, 2000). To encourage user participation, WIDE aims to provide design interactions in a controlled customization process using information filtering as the key mechanism.

WIDE takes the stand that non-designers design by searching through the space of design solutions, which are prepared by designers. From this perspective, two issues are key to the development of WIDE: first, the construction and contents of this space of design solutions, and second, the human-computer interaction to support this searching process. We propose to use a component-and-assembly database as the design space, and to adopt

information filtering (i.e., database query) as the search mechanism. However, the information filtering mechanism should be provided through a graphical interface where users can see the design result at all time. We plan to offer three levels of design interaction:

1. Apartment unit and layout arrangement selection: users can look for suitable apartment units according to their preferences, and can select different types of pre-designed layout arrangements that are suitable for a specific apartment unit and visualize the result.
2. Interior finishes/equipment selection: users can tryout different options of finishes and kitchen/bathroom equipment; and see the result of selection through computer rendered still images or virtual-reality walkthrough.
3. Advanced layout adjustment: users can customize interior layout for special needs through an intelligent design aids. For each confirmed change, the system visualizes the result in various ways to help users make their decisions.

There are two implementations of WIDE. WIDE-Kindom is the very first prototype. It is specially tailored for a specific apartment building project. The component-and-assembly database of WIDE-Kindom contains components (such as walls, rooms) and assembly tables, which record feasible component assemblies (such as alternative layout arrangements of an apartment unit). These components and assemblies are created upfront manually as well as through a set of macro commands based on design solutions by architects and modification guidelines (that allow architects' original designs to be modified to a certain extend).

WIDE-Kindom prototype implements the first two levels of design interaction: it allows users to identify suitable apartment units according to size, price, and Feng-Shui considerations and to customize selected units through various means. The results of user interaction in WIDE-Kindom are results of the information filtered through the component-and-assembly database. Figure 1 and 2 illustrate how design solutions change when a user identifies different design concerns.

Figure 1. An example of using wall finishes and floor surfaces as filters in WIDE-Kindom.

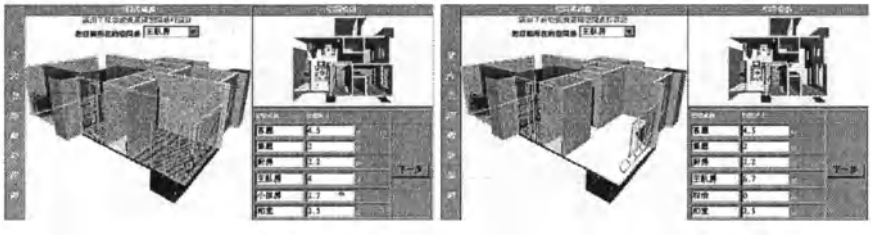


Figure 2. An example of using room combinations as filters in WIDE-Kindom.

WIDE-Roadhouse is the second prototype. It is specially tailored for roadhouses, which are typical in rural areas of Taiwan. During the *921 earthquake* in 1999, many roadhouses were destroyed. WIDE-Roadhouse prototype is set up to allow users quickly evaluate possible designs for rebuilding. The component-and-assembly database of WIDE-Roadhouse is similar to that of WIDE-Kindom. The knowledge encapsulated in this prototype is a roadhouse building system with improved structural performance and constructional efficiency (see Figure 3).

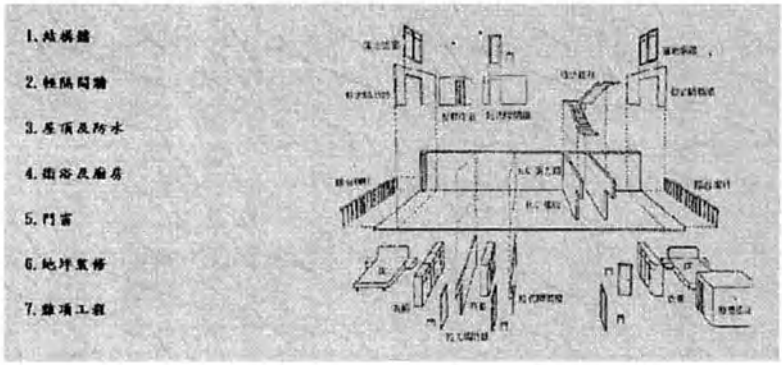


Figure 3. Components in the roadhouse building system
(source: Easylines Building System Co. Ltd.).

WIDE-Roadhouse prototype implements the first and third levels of design interaction: it allows users to identify suitable roadhouse configuration according to orientation, lot size, and primary functions; and to customize selected rooms through various means. Figure 4 illustrates how a user may examine and modify the interior arrangement of selected rooms.

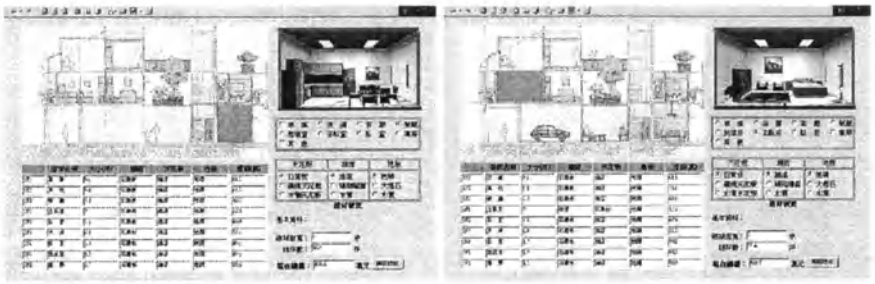


Figure 4. A sample of using room-function assignment as a filter in WIDE-Roadhouse.

Both WIDE-Kindom and WIDE-Roadhouse prototypes were exhibited in the "Construction and Automation 2000" exposition in Taipei. During the weeklong exposition, these prototypes were tested by visitors and received encouraging comments.

5. DISCUSSIONS

Compared to architects or designers, laypeople do not have design expertise, yet they need to make design decisions based on their individual needs, which may be quite different from person to person. The information filtering/search driven approach taken by WIDE provides an environment with encapsulated design knowledge (in the component-and-assembly database) and enables laypeople to make proper and reasonable design decisions. Although many do-it-yourself home design software applications (e.g., 3D Home Architect) allow laypeople to design, our approach aims to address some common limitations that are inherent in these applications.

1. design drawing as the means of communication: most of these applications assume laypeople understand design drawing (e.g., plans, elevations, sections) has use it as the primary form of communication, while we consider people should be able to design without having to understand design drawings (and in fact many people could not read design drawings) and the drawings should only be a kind of outputs produced by the design environment.
2. fixed design knowledge base: these software applications are tailored to specific types of designs (e. g., the design knowledge encapsulated in many applications is only suitable for use in North America but not for Taiwan), whereas our approach establishes a framework for design knowledge encapsulation which is not based on a particular building type nor a specific locality.
3. closed environment: these applications use proprietary data format that do not allow easy information access; our approach, however, is based on

the concept of information sharing and uses open network information exchange standards.

Our experience gained from implementing two WIDE prototypes has suggested that the information filtering/search driven approach has a promising future. The user interface designs of these two prototypes have borrowed concepts from primitive GIS systems. We plan to employ advanced information filtering mechanisms to support dynamic user interactions. Furthermore, this approach requires architects to design a family of solutions rather than one solution for a design problem. For the two prototypes, we have been working with architects who are familiar with the open building concept and specialized in architectural systems. Even after architects have designed families of solutions, however, constructing a component-and-assembly database (i.e., encoding design knowledge) is time-consuming, and will be a bottle-neck when we scale up the system. We are planning to employ generative mechanisms to create contents of the component-and-assembly database on the fly while users are interacting with the system. We hope to report the progress on these issues in the conference.

6. ACKNOWLEDGEMENTS

WIDE project is supported by Architecture and Building Research Institute of the Minister of Interior Affairs in Taiwan, Easylines Building System Co. Ltd. and Kindom Construction Co. Taiwan. Authors would like to acknowledge Te-Hsiang Yang, Sheng-Kai Hsu, and Chi-Chuan Lin who participated in the system implementation.

7. REFERENCES

Archea, J., 1987, "Puzzle-Making: What Architects Do When No One is Looking", in: Kalay (ed.), *The Computability of Design*, Wiley, New York, p. 37-52.

Borghoff, U.M. and R. Pareschi, 1998, *Information Technology for Knowledge Management*, Springer, New York.

Chien, S.-F. and S.-G. Shih, 2000, "A Web Environment to Support User Participation in the Development of Apartment Buildings", in: *Special Focus Symposium on WWW as the Framework for Collaboration, InterSymp 2000*, July 31-August 5, Baden-Baden, Germany, p.225~231.

Gomez, N., 1998, "Conceptual Structural Design Through Knowledge Hierarchies", Ph.D. Thesis, Department of Civil and Environmental Engineering, Carnegie Mellon University, Pittsburgh, PA.

Habraken, N.J., 1972, *Supports: An Alternative to Mass Housing*, Praeger, New York.

Habraken, N.J., 1998, *The Structure of the Ordinary: Form and Control in the Built Environment*, The MIT Press, Cambridge, MA.

Karlgren, J., K. Hook, A. Lantz, J. Palme, and D. Pargman, 1994, "The Glass Box User Model for Filtering", in: *4th International Conference on User Modeling*, Hyannis, ACM.

Newell, A. and H.A. Simon, 1972, *Human-Problem Solving*, Prentice-Hall, Englewood Cliffs, NJ.

Shih, S.-G. and S.-F. Chien., 2000, "Developments Towards Open Building in Taiwan: Midterm Report", Research Report, Architecture and Building Research Institute of the Minister of Interior Affairs in Taiwan, Taipei. (In Chinese)

Baptism of fire of a Web-based design assistant

Ann Heylighen and Herman Neuckermans
Katholieke Universiteit Leuven

Key words: design studios, utilization of Internet, design support, Case-Based Design

Abstract: DY*N*AMO – a Dynamic Architectural Memory On-line – is a Web-based design assistant to support architectural design education. The tool is conceived as an (inter-)active workhouse rather than a passive warehouse: it is interactively developed by and actively develops its users' design knowledge. Its most important feature is not merely that it presents students with design cases, but that those cases trigger in-depth explorations, stimulate reflection and prime discussions between students, design teachers and professional architects. Whereas previous papers have focused on the theoretical ideas behind DY*N*AMO and on how Web-technology enabled us to translate these ideas into a working prototype, this paper reports on the prototype's baptism of fire in a 4th year design studio. It describes the setting and procedure of the baptism, the participation of the studio teaching staff, and the reactions and appreciation of the students. Based on students' responses to a questionnaire and observations of the tool in use, we investigated whether DY*N*AMO succeeded in engaging students and what factors stimulated/hampered this engagement. Despite the prototype nature of the system, students were noticeably enthusiastic about the tool. Moreover, DY*N*AMO turned out to be fairly 'democratic', in the sense that it did not seem to privilege students with private access to or prior knowledge of computer technology. However, the responses to the questionnaire raise questions about the nature of students' engagement. Three factors revealed themselves as major obstacles to student (inter-)action: lack of time, lack of encouragement by the teachers and lack of studio equipment. Although these obstacles may not relate directly to DY*N*AMO itself, they might have prevented the tool from functioning the way it was originally meant to. The paper concludes with lessons learned for the future of DY*N*AMO and, more in general, of ICT in architectural design education.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 111–124.
© 2001 *Kluwer Academic Publishers.*

1. INTRODUCTION

DY*N*AMO, which stands for Dynamic Architectural Memory On-line, is a Web-based design assistant to support architectural design education. The tool tries to kill two birds with one stone. At short notice, it provides student-architects with a rich source of inspiration, ideas and knowledge for their studio projects, as it is filled with a permanently growing on-line collection of design cases. Its long-term objective is to initiate and nurture the life-long process of learning from (design) experience as suggested by the cognitive model underlying Case-Based Reasoning (CBR).

Whereas previous papers have focused on the theoretical ideas behind DY*N*AMO (Heylighen and Neuckermans, 2000a) and on how Web-technology enables us to translate these ideas into a working prototype (Heylighen and Neuckermans, 2000b), this paper reports on the prototype's baptism of fire in a 4th year design studio in Winter/Spring 1999. After a brief summary of the former papers, we will sketch the setting and procedure of this baptism, followed by the evaluation of the prototype's performance. Finally, conclusions are drawn for the future of DY*N*AMO and, more in general, of ICT in architectural design education.

2. A DYNAMIC ARCHITECTURAL MEMORY ON-LINE

CBR is a relatively young theory and technology within the field of Artificial Intelligence based on an alternative view of human reasoning. Rather than linking abstract pieces of knowledge (e.g. rules or models), reasoning is seen as remembering one or a small set of concrete instances and basing decisions on comparisons between the new situation and the old instance (Kolodner, 1993). This cognitive model was inspired by Roger Schank's Dynamic Memory Theory (Schank, 1982) and in turn inspired us to develop DY*N*AMO, a Dynamic Architectural Memory On-line (Heylighen, 2000).

Central to CBR's cognitive model is the claim that human memory is dynamically changing with every new experience (Schank, 1982). Furthermore, the model proposes some specific kind of changes: acquiring new cases, re-indexing cases already stored, and generalising individual cases. DY*N*AMO incorporates this cognitive model and at the same time extrapolates it beyond the individual, so as to embrace and profit from several kinds of interaction that are crucial for the development and renewal of design knowledge. This should result in a design tool that both feels cognitively comfortable to (student-)designers, and offers them a platform

for exchanging insights and knowledge, in the form of cases, with colleagues in different contexts and at different levels of experience.

Therefore DY*N*AMO is conceived as an (inter-)active workhouse rather than a passive warehouse (Schank and Cleary, 1995): it is interactively developed by and actively develops its users' design knowledge. Its most important feature is not merely that it presents cases, but that those cases trigger in-depth explorations, stimulate reflection and prime discussions between students, design teachers and professional architects.

Figure 1. Screenshot of DY*N*AMO's user interface

Translating these theoretical ideas into specifications for a concrete tool, at least two preconditions must be fulfilled: a. The case base should be simultaneously accessible to users at different locations – student-designers in architecture schools and professional architects in design offices. b. The content and structure of the case base must be able to change dynamically as it is used. With these (and other) demands in mind, we have chosen a Web-based approach to build a first prototype, which consists of three major ingredients: 1. a collection of design cases – the actual memory content of DY*N*AMO; 2. an underlying database that organises and structures this memory; and 3. a user interface allowing to both consult and modify

memory. The memory content and structure are stored at the server side, the interface consists of a standard Web browser at the client side (see Figure 1).

3. SETTING AND PROCEDURE

3.1 Students and design assignment

All 48 subjects in the study (12 female and 36 male) were student-architects of the 4th year Architecture, Urban Design and Planning at the Faculty of Engineering at the University of Leuven. Having had three years of design education, they participated in the reuse studio as it is an obligatory subject in the curriculum. Ten of them chose to work independently, the remaining 38 split off into two-person teams. The studio was led by Professor Paul Van Aerschot, the titular of the theoretical course on reuse, assisted by three studio teachers.

The students who participated in this design studio could choose one out of two design assignments: either the conversion of a former fabric hall into a public library, or a reorganisation of and extension to their school, which is located in the 16th century castle of Arenberg. Both assignments focus on dealing with an existing building, be it in the form of a reuse or extension project. Students are required to consider the building from multiple perspectives, such as formal articulation, spatial configuration, construction and materials used.

The site of the first assignment consists of the former ateliers of the Belgian Railway Company near Leuven. On this site, three factory halls are to be preserved and converted into a market hall, a public library and an indoor sports hall. Students were asked to design the library, and to investigate possible connections between library and market hall.

The second assignment is to be situated in the Arenberg park, where one campus of our university is located. The castle, which acts as landmark in this park, houses the administration of the Faculty of Engineering as well as the Department of Architecture. The task was to reorganise and optimise the West wing of the castle (design studios, lecture rooms, secretary, photocopy room) and expand it with a reception hall, material museum and exhibition room.

Five solo designers and 13 two-person teams chose the library project, the others – 5 singles and 6 pairs – opted for the reorganisation of and extension to the castle.

3.2 Procedure

Before the start of the assignment, DY*N*AMO was supplemented with information (primarily plans, sections and photographs) on several significant reuse projects and libraries, some of which were suggested by the studio teachers. Originally, we hoped that studio teachers would actively participate in loading the case base with relevant cases, yet simply getting them to make a list of projects already required quite an effort. Instead of the teachers, we collected information on these projects and added this information to DY*N*AMO's case base. At the start of the assignment, students were given a demonstration of how the prototype worked. All students participating in the studio were allowed to use the tool for as long and as often as they liked. By assigning each user – student or studio teacher – a user name and password, we could keep track of who logged on to the system and when. The system could be accessed not only from the computer class at school, but from every PC connected to the Internet. Students were able to consult the projects provided, write on-line comments, as well as add other cases they found relevant to the assignment. A DY*N*AMO help desk was set up to help students with using the tool, answer questions and respond to technical problems.

Aside from having access to DY*N*AMO, the design project was conducted as usual. Students met in the studio roughly twenty hours per week, spending much of this time either working alone or in pairs, or discussing their project with the studio teachers.

After the conceptual phase of the assignment, students were given a questionnaire in order to examine whether DY*N*AMO succeeded in engaging students. With an eye to future improvements, the questionnaire also asked how students liked several aspects of the tool, such as the interface, choice of cases or selection criteria. Some questions dealt with elements that may influence student engagement, such as whether or not students have a PC at home, like surfing on the Internet, or used CAD software to model their project. From the 48 students who participated in the studio, only 19 filled in the questionnaire. For this reason, the results in sections 4 and 5 are based on a sample of 19 students only.

Based on the students' replies to the questionnaires and logons registered by the system, the following sections will attempt to answer two questions: Did DY*N*AMO succeed in engaging students to explore the design cases provided? And what factors stimulated or hampered this engagement?

4. STUDENT APPRECIATION

From the 48 students who participated in the studio, 35 (73%) effectively made use of the system. Because the study described here represented DY*N*AMO's baptism of fire, we consider this percentage a reasonable level of success, all the more since using the tool was far from being stimulated by the studio staff. To the question whether they felt encouraged by studio teachers to use DY*N*AMO, 85% of the respondents answered in the negative. Only 11% had the impression that teachers were informed about the content of the case base. Judging from the registered logons, two of the four teachers had a look at the tool, be it only once or twice.

Unfortunately, the questionnaire, which was designed to examine whether students found the tool engaging, had much less success. As already mentioned, only 19 students (40%) filled in the questionnaire, which still represents a reasonable response given the strict time schedule of the assignment.[i]

4.1 Informal appreciation

The question whether they wanted to use DY*N*AMO again in future design projects, was answered positively by all respondents except for two. According to one of both, "it seems a very interesting tool, but we simply don't have time to use it." Rather than from lack of time, her colleague seemed to suffer from fear for design fixation, as he comments: "The chance of being original diminishes as information increases. EVERYTHING HAS ALREADY BEEN DONE. There are just some things that I'm not yet aware of."

When asked how they liked DY*N*AMO, it was evident from the answers that most respondents found the system engaging. The responses indicated that students were first of all attracted by the quality of the visual material and, to our surprise, by the tool's inherently 'ecological' nature. According to one third of the respondents, DY*N*AMO would finally offer a solution to the waste of paper caused by making copies in the library. Other strong points frequently popping up in the responses are the colour interface, the variety and scope of information and, compared to the library, the quick and easy way to find projects of interest. One, presumably extremely disorderly student especially appreciated the fact that, unlike paper copies, DY*N*AMO can never be lost.

The responses also indicated that, although students found the tool engaging, they were not necessarily engaged in ways that met with our expectations. In principle, we expected that students would consult cases in DY*N*AMO, but also actively participate in adding and commenting on

projects. Yet, none of the students exploited the opportunities for active participation. One student seemed to find the idea of 'information *from* students *for* students' quite compelling, even to the extent that he devoted a great deal of attention to praising it in the questionnaire. However, this same student then showed little or no interest in putting this idea into practice. This goal was probably too ambitious given the very limited time schedule of the assignment.

Asked about the drawbacks of DY*N*AMO, no less than 13 of the 19 respondents mentioned the obvious fact that you need a computer to consult the system. Despite serious efforts of our university, students do not always have easy access to computers, let alone to the Internet, and this may have prevented DY*N*AMO from functioning the way it was originally intended to. As one student put it: "If, during design, you just want to check something about a façade, a joining detail, a certain layout, then it's much simpler to consult the paper copies lying next to you on your table, than to wait till the next day to bike through the rain to the computer class." Yet, even for well equipped students, the medium by which DY*N*AMO makes cases available seemed to function as a brake: "It's difficult to go through during spare moments, say a train journey." However, the very fact that this student thinks of consulting the tool on the train suggests that he appreciates rather than questions its usefulness. Apart from the medium, respondents apparently had little to complain about. One student seemed disappointed by the number of cases in the system: "Once there are more projects, I'll come and look again," another complained about the fact that "you always have to press the submit button."

Finally, it should be noted that, to our surprise, students' responses were hardly affected by the prototype nature of the software. Most errors, such as JavaScript error messages, did not appear to disrupt students. However, they were occasionally frustrated by the slowness of the computers in the classroom, puzzled by platform-incompatibilities or confused by browser-dependencies. All in all, the DY*N*AMO helpdesk received five e-mails and one phone call.

4.2 Formal appreciation

In addition to an informal appreciation, students were asked how they liked specific aspects of DY*N*AMO by rating them on a five-point scale. Students could choose from *very poor*, *poor*, *neutral*, *good* or *very good*. Their responses were coded from 1 to 5, with *very poor* corresponding to 1 and *very good* to 5. The frequency and mean scores of their responses are displayed in Figures 2 to 4. Mean scores that differ significantly from a neutral 3 score in a paired *t* test are indicated by an asterisk (*).

Given DY*N*AMO's ambition to assist conceptual design, the questionnaire asked to what extent students felt supported by the system in this stage of the design process. On average, student opinions about the support for exploration, concept generation and concept development fluctuate around a neutral 3 score: 3.17, 2.94 and 2.88 respectively (Figure 2). None of the means are significantly different from 3 in a paired *t* test.

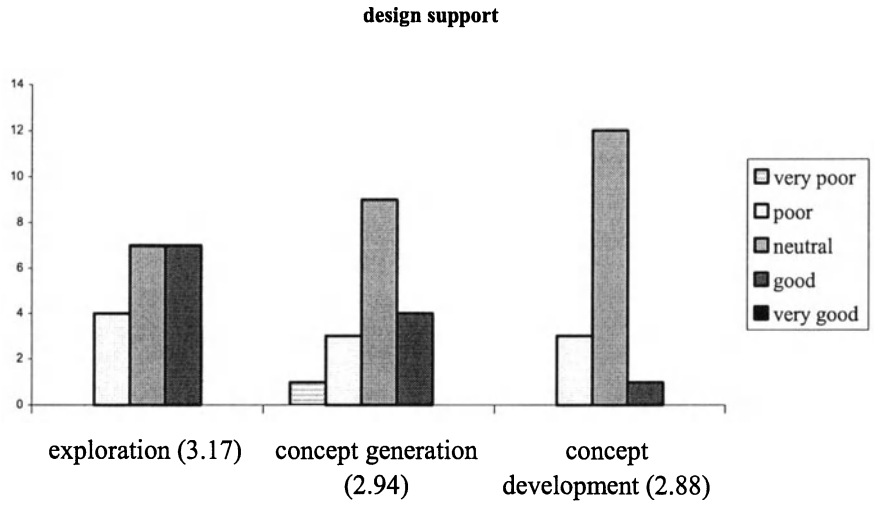


Figure 2. Frequency of student responses to the question whether they felt supported by DYNAMO during conceptual design. Mean scores are mentioned between brackets

Obviously, the extent to which DY*N*AMO can support designers is closely related to the quality of the case base. As for any CBD tool, its effective performance largely depends on the richness, diversity and number of cases (Oxman and Shabo, 1999). Asked about DY*N*AMO's case base, students seemed to be enthusiastic about the choice of the cases and selection criteria, yet remained neutral as to the case content (see Figure 3).

As indicated by the asterisk (*), the means for case choice and selection criteria – 3.56 and 3.61 respectively – are significantly higher than 3 ($p < .05$), whereas for case content – 3.28 – no significant difference is found. The positive rating for selection criteria chimes with the fact that students did not exploit the opportunity to create new indices. When asked why they did not, the majority of the respondents replied that DY*N*AMO's criteria were OK. Only one respondent suggested to add a criterion, or rather to do away with multiple criteria. He would prefer to simply select cases by typing in a keyword – whether this word pertains to a project's concept, context or

whatever – and let the system search across different indices to find a matching case.

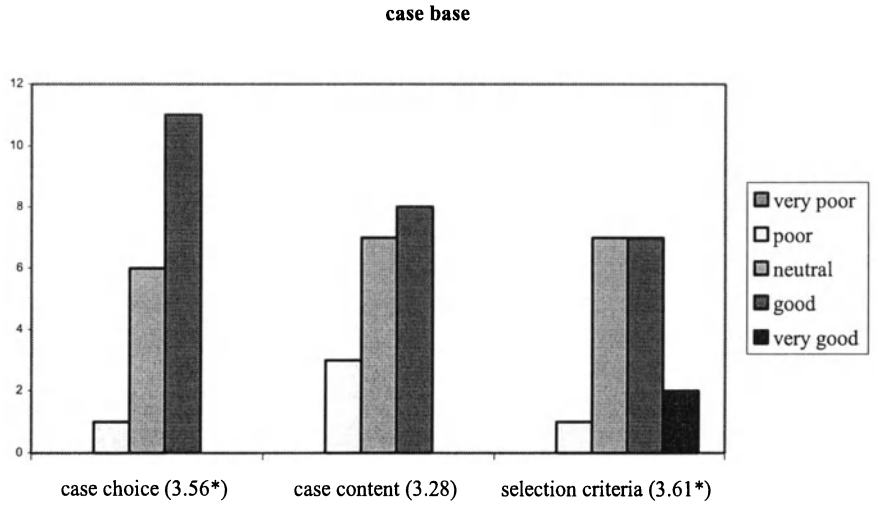


Figure 3. Frequency of student responses to the question how they liked specific aspects of DY*N*AMO's case base. Mean scores are mentioned between brackets and marked with an asterisk (*) when significantly different from the neutral 3 score in a paired *t* test ($p < .05$).

Finally, the questionnaire tried to find out whether the prototype was perceived as user-friendly. User-friendliness is a characteristic of every commercial software application, at least if we may believe their ads and salesmen. Although this characteristic is increasingly recognised as decisive for the success of an application, few people can describe what it actually consists of. Whether or not users perceive a programme as user-friendly turns out to depend on a cocktail of (sometimes subjective) factors (Froyen, 1999; Monden, 1999). Therefore, instead of asking for a general appreciation of DY*N*AMO's user-friendliness, the questionnaire solicited how students liked several ingredients of this cocktail: the interface, language used, programme speed, finding speed, ease of learning, ease of use and help contents. The interface of the prototype takes the form of a simple Web page, allowing users to interact by mouse clicks or typing. With an eye to using DY*N*AMO across architecture schools (and design offices) in different countries, the interface is written in English. The speed of the programme refers to how fast the system reacts to and processes user input, the finding speed to whether users can easily find the information they are looking for. An essential ingredient of user-friendliness is whether or not the programme

is easy to learn, without intensive study or months of unproductive bungling (Richens, 1989). Once the user has grown accustomed to the programme, it should be easy to use, without too much errors, serious (mental) effort or regular resort to the help button (Monden, 1999).

On average, students seemed either neutral or positive about DY*N*AMO's user-friendliness (see Figure 4). They liked the look and language of the interface, found the programme easy to use and were satisfied with the help contents. As indicated by the asterisk, the mean scores for these four aspects – 3.31, 3.65, 3.50 and 3.47 respectively – are significantly higher than 3 ($p < .05$). Programme speed, finding speed and ease of learning were judged neutral by the students, as no significant difference between the mean scores – 3.12, 2.83 and 3.38 – and the neutral 3 score was found.

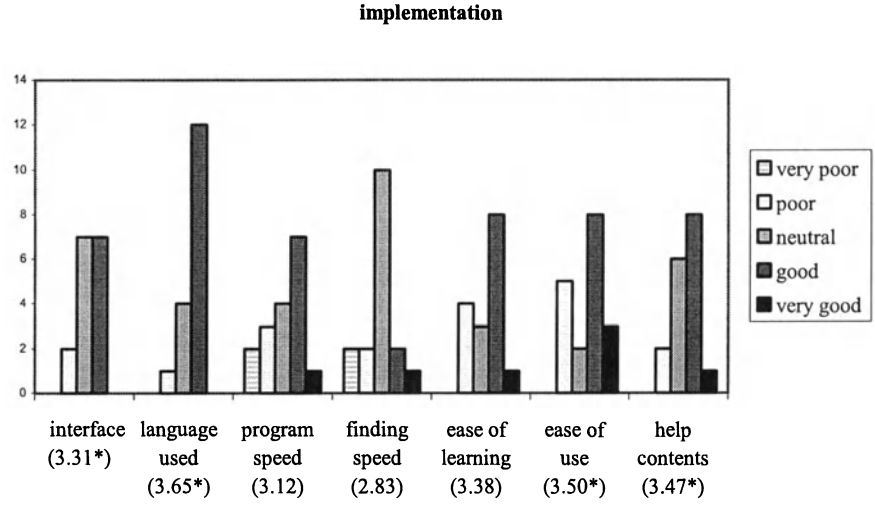


Figure 4. Frequency of student responses to the question how well they liked specific aspects of DY*N*AMO's user-friendliness. Mean scores are mentioned between brackets and marked with an asterisk (*) when significantly different from the neutral 3 score in a paired *t* test ($p < .05$).

In summary, there is significant evidence in students' responses that they found DY*N*AMO engaging. Three quarters of the participants effectively made use of the tool, and nearly all respondents would like to use it again in the future. Serious questions arise, however, about the nature of this engagement, since none of the students has taken up the offer to go deeply into cases. When asked how they liked specific aspects of the prototype, students were either neutral or positive. In particular, the case content, ease

of learning, programme and finding speed of DY*N*AMO received a neutral 3. Positive scores were given to the system's choice of cases and selection criteria, look and language of the interface, ease of use and help contents.

5. STIMULATING AND/OR HAMPERING FACTORS

Although DY*N*AMO managed to engage 35 participants in the studio, there remains a group of 13 students who could not be made to use the tool. Moreover, between the 35 DY*N*AMO users, there were considerable differences in terms of frequency of use. Judging from the logons registered by the system, some students had a look at the tool only once, others were logged on repeatedly. Therefore, the second objective of our evaluation was to find out what factors stimulated or hampered student engagement. The hypothesis being explored was that students who had access to the tool at home, are more computer-literate or used CAD software to model their design, would deploy DY*N*AMO more easily than others.

In order to examine this hypothesis, the questionnaire asked about students' personal computer equipment, how often they surf on the Internet, and whether they used a CAD package for this specific design project. Furthermore, we collected the scores students received for the CAAD course, which were used as a measure for their computer literacy. Subsequently, we calculated the correlations between, on the one hand, how often students made use of the system as recorded by the system and, on the other hand, the factors reflecting their computer equipment, familiarity with the Internet, use of CAD software and level of computer literacy.

The analyses of these correlations, however, yielded no results of statistical significance. The extent to which students made use of the prototype does not correlate significantly with any of the factors analysed. This seems to suggest that DY*N*AMO is fairly 'democratic', in the sense that it does not seem to privilege students with private access to or prior knowledge of computer technology. For the time being, the question why the tool did not manage to engage 13 of the 48 participants thus remains unanswered.

6. SUMMARY AND CONCLUSION

This paper has given an account of DY*N*AMO's baptism of fire. It has described how the system was used and appreciated in a 4th year design studio devoted to reuse. Based on students' responses to a questionnaire and logons registered by the system, an attempt was made to investigate whether

DY*N*AMO managed to engage students into in-depth explorations of design cases, and what factors stimulated or hampered this engagement.

Despite the prototype nature of the system, the answers to the questionnaire seem to suggest that students indeed found DY*N*AMO engaging. Questions arise, however, about the nature of this engagement. None of the students who used the tool exploited the opportunity to feed the case base, be it by writing an on-line comment, adding a project or indexing cases in a new way. Currently, DY*N*AMO's ability to engage students seems to come from the quality and colours of its visual material and the paper it saves, rather than from the opportunities for (inter-)action it offers. In part, the problem might be due to the strict time schedule of the assignment. Other possible explanations are the lack of enthusiasm on the side of the studio teachers, or problems of computer equipment. At the time of the assignment, the studio was not yet equipped with computers to consult DY*N*AMO, let alone with input devices (scanners, digital cameras) to add information. Although these obstacles – lack of time, of encouragement by the staff, and of equipment in the studio – may not relate directly to DY*N*AMO itself, they might have prevented the tool from functioning the way it was originally meant to. Actively encouraging students to consult and feed the case base and providing hardware devices to do so, could probably enhance the system's ability to engage students in (inter-)action considerably.

Other obstacles, however, do relate to DY*N*AMO itself and thus are to be charged on our own bill. Although students' were enthusiastic, they also have pointed out some aspects of DY*N*AMO that leave considerable room for improvement. Awaiting extra encouragement and computer equipment, the prototype was therefore significantly modified following this evaluation. First of all, we did away with the platform incompatibilities and browser dependencies of the system. Furthermore, the user interface was given a profound facelift to help users find their way more easily and quickly in the case base, and to make opportunities for (inter-)action immediately attract the eye.

The analyses to examine the question on stimulating or hampering factors, yielded no results of statistical significance. The extent to which students made use of the prototype does not seem to correlate significantly with personal access to or knowledge of computer technology, which suggests DY*N*AMO to be relatively 'democratic'.

Taken together, the evaluation of DY*N*AMO's first performance puts us in a hopeful mood concerning the future. There are, however, pitfalls in trying to evaluate the tool's performance. As already mentioned, its most important feature is not merely that it presents cases, but that those cases trigger in-depth explorations, stimulate reflections and prime discussions among students and studio teachers. The latter must help students understand

the relevance of retrieved cases through appropriate indexing and helpful on-line comments. Despite our firm intentions to actively involve the studio staff in the development of our prototype, we could hardly get them to have a look at the tool. As far as this involvement is concerned, we do not hesitate to say that DY*N*AMO's performances are more than below par.

Therefore, we would like to end this paper by stressing the need to adopt a broad(er) perspective when trying to assess the use of ICT applications in architectural design education. The effective performance of any such application does not only depend upon the sophistication of the software or the attractiveness of the interface, but also upon how it is introduced by teachers, perceived by students and supported by the school. If we are to develop tools that effectively support architectural design education, the influence of the context in which this education takes place should not be underestimated. Judging from DY*N*AMO's baptism of fire, to put dynamics into a case base is one thing, to 'dynamise' the context it is meant for is yet something different.

7. ACKNOWLEDGEMENTS

This research is sponsored by the Fund of Scientific Research (FWO) Flanders, of which Ann Heylighen is a Postdoctoral Research Fellow. The authors would like to thank the students, titular and teachers of the 4th year design studio for their collaboration. A very special thanks goes to Dr. Ilse Verstijnen for her invaluable help with the evaluation.

8. REFERENCES

Froyen, D., 1999, *Efficiëntie van CAAD-programmatuur. Gebruiksvriendelijkheid en mogelijkheden*, graduate's thesis, K.U.Leuven - Dept. ASRO, Leuven.

Heylighen, A., 2000, *In case of architectural design. Critique and praise of Case-Based Design in architecture*, PhD thesis, K.U.Leuven - Faculty of Applied Sciences, Leuven.

Heylighen, A., and H. Neuckermans, 2000a, "DY*N*AMO: Dynamic Architectural Memory On-line", *Educational Technology and Society,* 3(2), p. 86-95.

Heylighen, A., and H. Neuckermans, 2000b, "DY*N*AMO in Action: Development and Use of a Web-based Design Tool", in: Pohl and Fowler (eds.) *Advances in Computer-Based and Web-Based Collaborative Systems, Proceedings of InterSymp-2000 International Conference On Systems Research, Informatics and Cybernetics, Baden-Baden, Germany,* p. 233-242.

Kolodner, J.L., 1993, *Case-Based Reasoning*, Morgan Kaufmann, San Mateo (Ca).

Monden, P., 1999, *Analyse van de gebruiksvriendelijkheid van CAAD – programmavergelijking*, graduate's thesis, K.U.Leuven - Dept. ASRO, Leuven.

Oxman, R., and A. Shabo, 1999, "The web as a visual design medium", in *International Conference on Information Visualization*, IEEE Publication, London.

Richens, P., 1989, *MicroCAD Software Evaluated – A critical evaluation of microcomputer based CAD systems for the construction industry*, Construction Industry Computing Association, Cambridge (Mass).

Schank, R.C., 1982, *Dynamic Memory – A Theory of Reminding and Learning in Computers and People*, Cambridge University Press, Cambridge (Mass).

Schank, R.C., and C. Cleary, 1995, *Engines for education*, Lawrence Erlbaum, Hillsdale (NJ).

---

[i] Apart from lack of time, another factor that may explain this moderate response rate is the fact that students were asked to complete the questionnaire by the 1st author, a research assistant not formally involved in the studio, who has obviously less authority than the design teacher(s) in charge of the assignment.

On the narrative structure of Virtual Reality walkthroughs

An analysis and proposed design

Pieter Jan Stappers, Daniel Saakes, and Jorrit Adriaanse[1]
Delft University of Technology
[1] *SARA Computing and Networking Services*

Key words: Virtual Reality, User-centered Design, Narrative theory

Abstract: Architectural walkthroughs have often been presented as prime examples of applications that can benefit from Virtual Reality (VR) technology, but still VR presentations can be disappointing. A main reason for this is that most VR applications have been developed on purely technical criteria, with an emphasis on geometrical precision rather than experiential quality. In this paper we present a human-centered analysis and propose design solutions, by focusing on the narrative aspects of the walkthrough, such as connecting transitions of the kind used in contemporary computer games. The solutions show how such narrative enhancements can improve the user's experience of presentations at modest technical expense.

1. WALKTHROUGHS IN VR PRESENTATIONS

Architectural walkthroughs have been among the most cited applications of Virtual Reality (VR), and are becoming a regular feature in the presentation of large architectural design projects. As always, such presentations are partly show, partly evaluation. Just as design sketches, VR presentations can elicit constructive criticism from clients and users who find it hard to read the formal language of plan views and construction drawings. The power of Virtual Reality (VR) presentations lies in its direct appeal to the senses.

Nevertheless, VR presentations can be disappointing as experiences. The audience is impressed by being immersed in a 3D animated environment, but notices that the graphic quality of VR is lesser than that of static

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 125–138.
© 2001 *Kluwer Academic Publishers.*

renderings that can be made with a down-to-earth PC or workstation. Also, the simulated environment appears dead and empty, devoid of life which architects can readily suggest in conventional media, but which are hard to program in VR. The audience can see spectacular events, such as flying through space, but have no direct interaction with the environment: a guide controls the input device.

These criticisms are not a necessary feature of the technology. In another area of simulation, 3D computer games, the integration of technology and user-centered design has produced compelling experiences on relatively modest machines. Especially aspects of narrative are better represented, and some of these techniques can be fruitfully applied in VR walkthroughs as well. In this paper, we use narrative theory to analyse the structure of existing walkthroughs in the CAVE, compare this to recent computer games, and draw conclusions in the form of solutions for a user-centered design for architectural walkthrough presentations.

1.1 Role of the walkthrough in the design process

Figure 1 depicts an architectural presentation in the CAVE, a 3*3*3 meter cubic space in which small audiences can take part in a Virtual Reality experience. On its walls (in the Amsterdam CAVE three walls and the floor) stereo images are displayed whose perspective matches the position of a 3D sensor on one user's head. For that user, views in all directions correspond to a 3D environment. Other users close to this prime user, receive stereo images which appear a little distorted, but still give a compelling spatial impression. In this way, small groups of users can be led around and through simulated buildings.

A CAVE walkthrough is typically used for final presentations of large building projects. Participants in these presentations, e.g. staff of municipal councils or project developers, are often less trained in reading construction drawings, and benefit from experiencing the design in a 1:1 immersive simulation. The CAVE supports communication between the participants, also because the participants can see each other (which is not the case in helmet-mounted VR). Because multiple CAVE systems can be digitally coupled, the same environment can be shared by teams at separated geographical locations.

*Figure 1.*Participants in a CAVE presentation wear stereo goggles to see stereo images on four wall displays. In a typical architectural walkthrough, an architect and a VR specialist guide a group of people representing the client through parts of the proposed building.

The model of the proposed building that is viewed in the CAVE, is usually derived from a CAD model that had been made earlier in the design process. This derived model (which is often simplified to fit the needs of real-time visualisation) can be traversed by means of navigation software. It is important to note that the 3D session is not an isolated product, but plays a part in a larger presentation.

1.2 Worst-case scenario

We illustrate the shortcomings of many current presentations in a worst-case scenario. This scenario is exaggerated, but we have seen all its elements in various demos in various labs. The scenario is not meant to criticize the makers involved, but to indicate aspects which are easily overlooked because they are often thought as 'not a technical part of the simulation'. Figure 2 outlines the sequential structure of the scenario.

Before the simulation has started, the audience, filled with anticipation because they have been told to expect 'the wonders of VR', enter a heavily wired room in the basement of a computer center, where technicians are just calibrating the equipment or making last fixes to programs. Then they go through a confusing ritual of putting on stereo goggles, the light is turned off and the VR displays are filled with command line displays showing technical data.

After a warning by the guide, a 3D world suddenly pops up around the audience, and the guide flies them through the world. Because interaction is still a cumbersome part of most VR applications, the audience can only look around, while the guide controls movement through using a spatial joystick. Although the 3D view is impressive, the image quality is less than that of the renderings of the building that they saw before. The audience quickly finds out that they cannot easily indicate parts of the scene by pointing with their hands. Perspective distortions in the CAVE make it impossible for one person to judge what direction another person is aiming at (CAVE perspectives are only geometrically correct for the single user who is wearing the tracking sensor). Moreover, the simulated building appears

static, dead, clean, straight from the CAD package, and participants find it hard to orient themselves in the building. After the guide has shown the different parts of the building, the world disappears as suddenly as it came up.

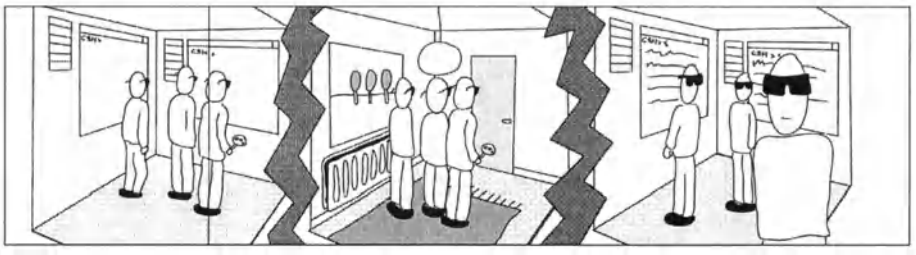


Figure 2. In the worst-case scenario, the simulation is sandwiched abruptly between screens with technical data which lessen the quality of the VR experience.

After the simulation ends, the audience is slowly ushered out of the room. The CAVE contains nothing to remind them of their walkthrough. Although the audience has seen the geometry, their experience might have been much more compelling.

Some VR sessions continue with a few spectacular demos in order to impress and amuse the guests. These demos can provide an unintended counterpoint to the walkthrough itself, because they contain graphics and interaction that was designed and tuned to impress the audience, and therefore are experienced as 'much better than the walkthrough', creating a sense of disappointment about the architectural presentation itself.

1.3 Reality check

The scenario sketched above does not do justice to the current state of the art, but all of the elements can be found in current practice around the world. In our own practice we found

- people were not able to recognise the VR model of the existing building in which they worked. The simulation lacked environmental detail and spaces filled with people.
- people were disappointed by the graphics and interaction. Their expectations had been raised by special effects in games and movies.
- a lack of support for the architect giving the actual presentation.
- the walkthrough not adding enough value to the presentation to justify the high costs involved.

In general the VR walkthrough lacks the interesting interactive techniques of VR and also lacks the richness and expression of traditional

presenting methods. Some of the techniques which we will discuss below are used in some demonstrations at SARA, but are not standard practice.

In this section we have outlined some limitations of current VR practice: although the graphics can be impressive, the user experience is often found lacking. This should not be surprising, since VR originated in computer graphics laboratories, with the addition of richer input devices, but only more recently has it received interdisciplinary user-centered study and design.

2. THEORY: NARRATIVE STRUCTURE

Narrative theory was popularized in the field of computer-human interface design by Brenda Laurel with her book 'Computers as Theatre' (1993). The theory has widely been praised, but real applications of the theory are hard to find. Virtual Reality walkthroughs can be an excellent case study, exactly because the presentation of a building is a form of storytelling, of engaging an audience into an experience.

Here we distinguish three narrative elements. First is the role of time. Many VR simulations consist of placing the user in a piece of simulated geometry, and allowing him or her to fly through it for the duration of the experience. But people's interaction with products is not an iteration loop. Just like a book or a movie, it has a beginning, a middle, and an end, and sometimes an intricate structure in-between.

Second, the role of space. The world in which we live is littered with traces of history, little details which help us distinguish where we are, what we and others have done, and what we can do in the future. Most VR simulations are empty, and present a geometrical world which is experienced as 'dead', 'expressionless', 'too abstracted to be real'.

Third, the role of action. In the natural world, and in a play, we act upon the world by using our body, not by issuing commands. For example, we open a drawer by pulling it with our hand, not by speaking to it, either by voice or through a remote-control device.

In this section we illustrate how these roles have been used to heighten experience and engagement by the makers of 3D computer games. These games, which emerged in a highly competitive and experience-oriented market, can teach us 'tricks of the trade' which can be used to improve the experiential quality of VR simulations.

2.1 Time: Sequential structure in computer games

Figure 3. Advertisements, packaging, cinematics and gameplay together enhance the game experience in computer games such as Core Design's 'Tomb Raider'.

The experience of a computer game player starts long before the game titles appear on the computer screen. In the case of 'Tomb Raider', the main figure was billed in the way of a movie star, long before the game could be bought. People who bought it first heard about it from an orchestrated press campaign, then went through the ritual of opening the (carefully designed) box.

When the game commenced, the actual gameplay was preceded and interspersed with cinematics, prerendered animations which are of much higher graphic quality than the interactive part can be. An introduction provided a background (date and place of birth, social class, etc.) to the main figure and storyline and atmosphere that extended far beyond the jump, search, and shoot actions of the game itself. Interestingly, players often remember the visual quality of the gameplay as that of the cinematic scenes.

At the end of the game, the player has to combat the dragon in the final confrontation. When he or she wins, a special cinematic 'reward' fragment follows. In the past years, several papers in the professional press have addressed the structure of the storylines in this kind of games (e.g., Costikyan, 2000).

2.2 Space: environmental narrative

In many respects, it's the physical space that does much of the work of conveying the story the designers are trying to tell. Color, lighting and even the texture can fill an audience with excitement or dread (Carson, 2000). Objects in the environment provide feedforward: keys lying about and footstep trails of footsteps on a floor inform the player of interesting new locations, and how to get there.

2.3 Action: enactment versus indication

Despite the limitations and diversity of the input devices that the games must support (keyboard, mouse, joystick, and game paddle must all work), the player is tied to the environment through actions tuned to the tasks in the game. He or she acts upon objects instead of giving disembodied verbal or key-press commands according to an arbitrary convention.

2.4 Conclusion

Many of the narrative elements do not affect the technical performance of a simulation. Many of the elements take place *outside* the simulation itself, but carry a psychological enhancement into it. The same elements can be used to improve VR simulations. This does not mean that architectural walkthroughs should be made into computer games. But it does suggest that some narrative devices may be used to enhance the user's experience of the proposed architecture. In the final parts of the paper we discuss how simple solutions in the setting, the hardware, and the software can improve the look and feel of existing walkthroughs, and better integrate the Virtual Reality simulation within the larger structure of the architectural presentation.

3. APPLICATION: NARRATIVE ELEMENTS IN WALKTHROUGHS

In this section we discuss the architectural simulation within the framework of narrative theory.

3.1 Sequential structure

The transition from the real world into the virtual environment is often severe. As the guests enter the hall which contains the CAVE, the CAVE's walls displays double images in distorted perspective, because the sensor and stereo goggles are hanging from the ceiling. Once the tracked user wears his goggles, the 3D environment can be experienced.

The second transition is the one that brings the user in the surroundings of the building, or into its interior. The route in the environment reflects the story that the architect wants to present. In our own experience we found that some architects want to start with an overview, while others start with details gradually exposing more and more, to create tension and to set expectations.

The route shows the rhythm of the building, the contrasts between small and large spaces, and is the key benefit of VR compared to the traditional presenting methods. The walk through the building conveys a much richer experience of it, than can be had from viewing a number of vistas. A carefully planned route, can make spaces look bigger, lighter etc. Currently, the route and its transitions are all implicit, 'in the head' of the guide, *enabled* but not *supported* by the simulation. In section 5 we propose a few solutions to support the guide in telling the story.

3.2 Narrative in the environment

The task of the guests in a presentation is to look around and understand the building, and we want to make this limited task interesting by creating a rich, self-explaining environment. Let the users discover the function and atmosphere of spaces themselves, rather than having the guide tell them what to see or feel. If the simulated building is empty of furniture, it is hard to imagine it as rooms that are used. Simple furniture gives cues about the scale, but what's more important it explains by whom and how the room is used. Props and lighting direct the guests' attention, and allow the guests to indicate places (to which they cannot point) by naming them, e.g., 'the door next to the painting'.

All the elements that define a room, the ambience and cues must be added to the visualisation. In immersive VR we need to explicitly show things which in sketchy renderings can often be left to the imagination.

A special case is adding people, for scale, to denote a room's function, to depict its atmosphere. In drawing classes all architects have learned quick and sketchy methods to add people and furniture to visualisations. In VR there is a difference between images of furniture and people, in that the VR users are part of the visualised scene, and therefore expect people to react to their presence. In the Alladin VR-ride, Pausch et al. (1996) found that users did not mind that simulated agents were rendered in cartoon-style, but were disturbed by the fact that the agents did not react to the presence of the users. When we populate our architectural simulation with simulated people, we must make sure these react at least in some way to the proximity of the guide and guests.

3.3 Enactment

The example just given of simulated actors was already an example of the role of action. In current presentations, the guests have no direct interaction with the environment. All control over the presentation is in the hands of the guide. An experienced guide in the demonstrations at SARA

can give the user the needed feedforward by announcing that interesting things are about to appear around a corner. The anticipation thus created enhances the experience of the audience.

One way to enhance the experience is to let the guide open a door by grasping its handle rather than pushing a button on the remote control. The grasping action is visible to all participants, and is experienced as a natural action in the context of going through a door. Although the guests do not act themselves, this action heightens the suggested connection between the virtual and real environments, and therefore heightens their sense of presence in the virtual environment.

3.4 Proposed solution: a better scenario

We conclude this section by presenting solutions in the form of a scenario. The added value of Virtual Reality systems is that the proposed design is not just told by the architect, but also that the guests can themselves experience, discover, and explore the design. The simulated spaces must tell their story, and enhance the limited task of the guests: looking around, pointing out interesting places or problems. Solutions to this problem involve enhancing the simulation in the direction of expressiveness, which need not be photorealistic modelling. In this section we present a number of solutions, none of which involve much technical effort, but try to improve the fit between the simulation and the users' needs on the basis of existing technology.

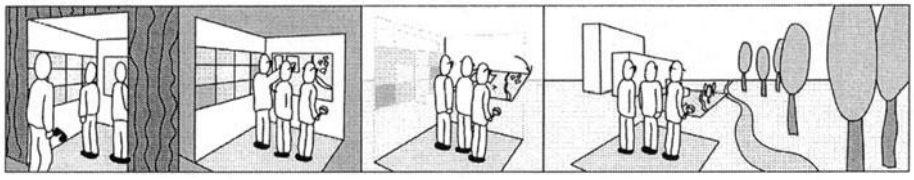


Figure 4. At the start of the proposed scenario, the CAVE displays introductory 2D presentation material about the proposed building project. When the 3D goggles are active, a smooth transition is made to 3D presentation, and the building is approached.

3.4.1 Before the simulation starts

The users enter the large space with the CAVE, the hall is dark, and the CAVE is light and attractive. The walls of the CAVE are filled with 2D images, animations and maybe music. These images provide the background story and the lit CAVE attracts the users to enter it. The imagery is displayed in mono, on the walls, so there's no perspective distortion or double images. While standing in the CAVE the users put on the goggles.

The architect starts presenting using the CAVE as a large, conventional slide display. This preparation ends with the plan view, the map of the building, depicted on one of the CAVE's walls. While the guide is talking and pointing at parts of the map, the simulation is loading.

3.4.2 Starting the walkthrough

The 3D experience begins when the guide 'pulls' the map of the building down from the wall, and it turns into a tabletop-size scale model, and the building can be discussed on model scale. The walls of the CAVE gradually disappear to create a smooth transition to the virtual environment. The model helps to smoothen the passage of the guests from the real environment into the virtual environment. It connects these environments, preventing the guests from becoming disoriented.

3.4.3 During the walkthrough

In the virtual environment there are simulated actors which support the route. The guide can 'hook' the navigation software to these actors, so the group follows the actor rather than flying through space. The actor provides feedforward by leading the way, strengthens the tie between the route and the virtual environment, and gives the movements a sense of purpose related to the story. So in our story the guests follow the actor to the building. The building is not shown in isolation, but other buildings surrounding it, are suggested by some form of outlines. The environmental context helps to set the mood. The simulated street environment contains the suggestion of life and traffic: cyclist and cars pass by, and react to the presence of the guests. A cyclist can look at the users, a car stops while the users cross the street.

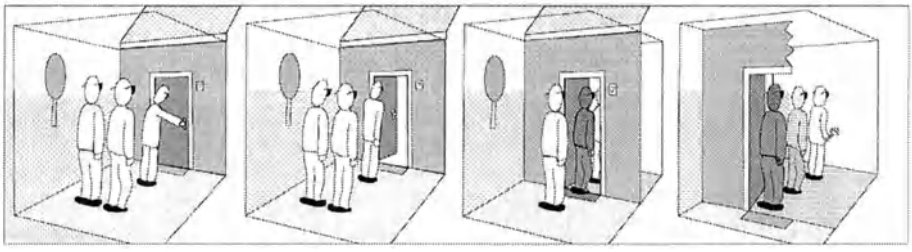


Figure 5. Opening ritual and light effects enhance the experience of entering the building. The building's wall is aligned with the physical display wall, so optical distortions are minimized.

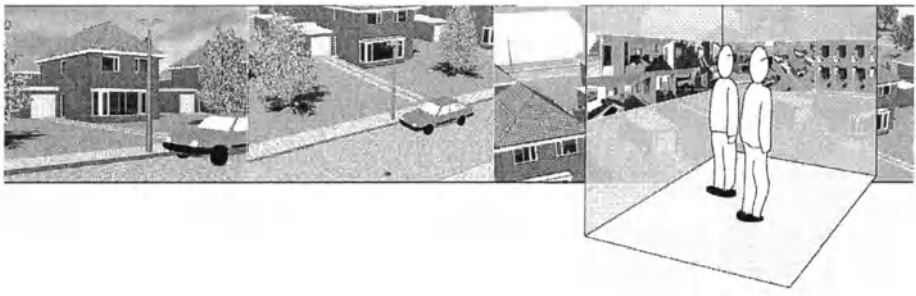


Figure 6. After the walkthrough, the viewpoint rises to an overview, and a connecting transition is made to 2D imagery showing highlights of the tour. and images of the places they visited. (Model courtesy of Zegelaar Onnekes BV)

When the group halts at the door of the building, a new transition occurs. The guide spends some time talking, then grasps to open the door and the users enter the building. We don't promote to always open all doors in such a ritual, but this ritual is important to stop and experience the entering of a building for the first time.

Two technical problems need to be overcome in this transition. First, in order to eliminate perspective distortions during the talking and grasping of the doorknob, navigation stops not just anywhere in space, but in such a location and orientation that the door and wall are aligned with the display walls of the CAVE (in that case, distortion is absent). Second, because the lighting model used in the CAVE's graphics software, typically indoor rooms are brighter than outside scenes, whereas in the real world a reverse relation holds. We can still give people the sensation of entering a building by explicitly modulating the overall light intensity at the moment they move into or out of the building. As the group enters, the overall light intensity is dropped quickly and noticeably, after that it increases again slowly and less noticeably. By tuning these transitions, the experience of entering a building is given both dramatic impact and a heightened degree of realism.

What we just described involves not only graphics but also sound. Environmental sounds can convey a strong sense of place, distance, and atmosphere (Mountford & Gaver, 1990). Sounds of traffic can easily be muffled or attenuated when the group enters a simulated building.

A building consist of important places to visit but also of less interesting spaces. We lead the users to the more attractive places by placing *attractors* to guide them, i.e., objects like furniture that draw the attention. Localized sound can also be used to draw the attention. In the less interesting spaces, movements can be increased a bit. Note that these additions do not take away functionality, but rather invite optimal behaviour. The guide can still move toward boring spaces or leave the route when he or she needs to.

The rooms are filled with simple cues which make the box-like simple geometry into rooms, cues like skirting boards and window frames, and dirt on the floor leading up to much-used doors. The aim in all these additions, however, is not to increase the supposed match between the simulation and a physical reality, but to increase the match between the simulation and the presentation's message (Stappers et al, 2001).

3.4.4 After the simulation

The walkthrough ends with the users rising on an elevator platform and thus get an overview of the building. Representative views of parts of the tour appear on the walls of the CAVE. Just as holiday snapshots, they remind the audience of the things they have seen, and help to reconnect the virtual environment to the real environment. These images draw the users out of the virtual environment, and the discussion continues, but now with an overview of the building and all the important images and details compressed in one space. Again the imagery is 2D, and the users can take off their goggles. When the talk is over, the users leave the CAVE and continue their talk in the conference room. There they may find the pictures summarizing the walkthrough in a printed form.

3.5 Partial conclusions

During this research we found that solutions which enhance the quality of the narrative (pauses, breaks, anticipation, and climax) can at the same time solve problems at the more mundane technical level of presentation. For instance, when presenting to more than one person in a CAVE environment, confusion often arises when someone points with a finger, because apparent distortions of perspective stereo lets different viewers perceive different directions of pointing. A solution is to place non-obtrusive, natural objects near points that are likely to be discussed, so they can be named. Similarly, objects placed beyond open doorways can be used to draw the viewer's attention, in the way some artists entice the viewer's eyes to move over their painting in a certain way, or the way in which objects lead the way in some maze-solving computer games.

3.6 Incorporating narrative elements in the development process of architectural presentations

The scenario sketched in the previous subsection relied on technically simple tricks to enhance the support for the narrative elements of the presentation. In themselves, these tricks are not difficult. The difficulty lies

in finding out what effects the hardware and software are able to produce (knowledge of the VR specialist), finding out what the presentation storyline needs (knowledge of the architect), and merging these knowledge into a well-tuned set of support effects. This requires communication between experts from different backgrounds, knowledge, and skills. In the current project, this merging was performed by an industrial design engineer. In the future, a formal structure is needed to support and promote the required communication between user-centered and technology-centered expertises.

A structured approach goes beyond a choice for a certain software package or library. No current software packages help the architect to see what narrative-sustaining effects the CAVE can perform, or what it costs to prepare them or to run them. And the technology changes rapidly. Neither can or should we turn VR specialists into architects. What is needed is finding a joined language for both parties.

One such language has been found in '(software) patterns', structured expositions of a problem, its importance, example solutions, and observations. This technique of describing, pioneered by Christopher Alexander (1977), and taken up enthusiastically by the software engineering community (e.g., Gebriel, 1996) and recently by the game developer community (Hecker & Simpson, 2000) can provide a common platform. In patterns, a concrete prototype plays the central role. All the discussants understand the prototype and can discuss it from their own perspective. Through the concrete nature of prototypes, a catalogue of techniques and parts can be grown. These may not be as formally unified as a software library, but are also more generally applicable.

The resulting effects can be used as a standard, shared, bag of tricks, which is enhanced and added to over time, and which helps in the specification of elements for new presentations.

4. CONCLUSIONS

In this paper we have argued how high technology VR systems in themselves do not guarantee highly convincing experiences for users. The level of experience and engagement in some commercial 3D games goes well beyond those of most academic and commercial VR presentations, for a large part because the former develop their applications from a user-centered criterion. We have shown how narrative theory, originally developed for theatre, but in the past decade also applied to product experiences, can be applied to enhance the experience users get from architectural VR simulations. All together, the results show how a high-end

VR architectural presentation can be improved by learning from the trade secrets of the game industry.

The proposed design solutions in this paper address only some aspects of a simulation, but indicate a practical way, through patterns and prototypes, through which narrative elements can be brought into the development process of a VR developer. Rigorous testing is still needed, and a suite of examples will have to be grown and tuned, before the solutions we present settle in the working practice of VR centers.

5. REFERENCES

Alexander, C. 1977, *A pattern language*, Oxford University Press, New York.

Carson, D. , 2000, Environmental storytelling ,*Gamasutra web archive,* http://www.gamasutra.com/features/20000301/carson_01.htm

Costikyan, G. (2000) Where stories end and games begin. *Game Developer*, Sept., 44-53.

Cruz-Neira, C., Sandin, D.J., & DeFanti, T.A. (1993) Surround-screen projection-based Virtual Reality: the design and implementation of the CAVE. *Computer Graphics: Proceedings of SIGGRAPH '93*, August 1993.

Gebriel, R.P. 1996, Patterns of Software, Oxford University Press, New York.

Hecker, C., & Simpson, Z.B. (2000) Game programming patterns & idioms. Game Developer, Sept., 12-13.

Laurel, B. , 1993, *Computers as theatre*, Addison Wesley Longman Inc. , City.

Mountford, S.J., & Gaver, W.W. (1990) Talking and listening to computers. In: Laurel, B. (Ed), *The Art of Human-Computer Interface Design*. Addison-Wesley, 319-334.

Pausch, R. , Snoddy, J, Taylor R. , Watson S. and Haseltine, E. , 1996, "Disney's Aladdin: First Steps Toward Storytelling in Virtual Reality", *Computer Graphics Proceedings, Annual Conference Series*, 23(4), p. 193-203.

Stappers, P.J., Gaver, W.W. , & Overbeeke, C.J., (2001) Beyond the limits of real-time realism. In: Hettinger, L.J., & Haas, M.W., *Psychological Issues in the Design and Use of Adaptive, Virtual Interfaces*. Erlbaum, New York

Towards a natural and appropriate Architectural Virtual Reality: the nAVRgate project

Past, present, future.

Michael Knight and André Brown
School of Architecture and Building Engineering
University of Liverpool

Key words: Virtual Environments, Navigation, Interaction, Perception

Abstract: The lure of virtual environments is strong and the apparent potential is enticing. But questions of how Human Computer Interaction (HCI) issues should be handled and married with best practice in Human-Human Interaction (HHI) remains largely unresolved. How should architectural images and ideas be most appropriately represented, and how should designers interact and react through this computer mediated medium? Whilst there is never likely to be unanimity in answer to such questions, we can develop new ideas and new systems, test them, report on them and invite comment.

The nature and novelty of virtual environments is such that refinements and innovations are likely to come from a variety of sources and in a variety of ways. The work described here explains the evolution and current plans for the development of a particular approach that has been developed and refined by the authors. Low-cost, effective and appropriate are the key words that have driven the developments behind the evolving nAVRgate system that has arisen from this work, and that is described here.

1. THE CONTEXT

The paper reports on the nAVRgate project which is now in its second phase. In the first phase a low-cost, but effective virtual environment system was developed. The software used was computationally efficient, and the physical interface was direct and effective. The keyboard and mouse were replaced by an interaction device that we believed was more familiar and

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 139–149.
© 2001 *Kluwer Academic Publishers.*

hence more appropriate to navigating through urban environments, the bicycle. In this respect the system has a lineage that can be traced back to the Legible Cities idea suggested by Jeffrey Shaw (1994). The nAVRgate-1 system has been demonstrated and reviewed. Following that review a second stage of development has taken place. The paper here will present a summary of the nAVRgate-1 system, the principal points arising from the review process and a description of the current stage of development, nAVRgate-2.

nAVRgate-2 brings together three strands of refinement. First, new ideas on low-cost, natural and appropriate interaction devices have been developed. Second, ideas on how the shortcomings of the software in nAVRgate-1, without the loss of computational efficiency, might be addressed have been investigated. Thirdly, ideas that have arisen from parallel work in the authors' research group on the nature and differences in perception of architectural images in a computer mediated environment are being drawn in. We aim to show that these ideas can bring positive enhancements to what was already an effective system environment.

The work described here emphasises the Navigation aspect of our Virtual environment. Sometimes navigation is confused with locomotion but, as Darken et. al (1998) note:

> 'Navigation must be seen as a process. We often make the mistake of seeing it as its end result – locomotion- navigation's most visible attribute. However, the cognitive subtasks that drive locomotion...are an integral part of the overall task'

Inextricably linked with the idea of navigation is the mental representation, the cognitive map, of our understanding of the environment. In other disciplines concerns over the visual representations of information in virtual environments raises new issues. But in architecture there is a legacy of visual representation in the real world that we need to examine and take account of in designing new virtual environments. Maher et. al. (2000) make the point:

> 'Although architectural design is noted for the forms and places created, the semantics of these places lies also in their function. The functional aspects of physical architecture can influence the design of virtual worlds'

Contemporary thoughts on how we represent ideas and the world around us can be found to be linked to contemporary technological developments. It is not simply a matter of coincidence that Freud's theories on dreams coincided with the birth of the technology that gave us films and cinema in the last century. That was a case of new technology inspiring creative

thought and a broader understanding of an issue. This century begins with the technology and philosophy that relate to Virtual Environments in their relative infancy. The work described in this paper is aimed at helping to move the associated debate along.

2. The nAVRgate project

As we mention above, our initial intention in the nAVRgate project (AVR being a sub-acronym for Architectural Virtual Reality) was to develop a low cost Virtual Environment where navigation was driven through natural locomotion. An initial desire was to open up the possibility of experiencing large scale urban environments to an audience; that is with a large display area and wide angle of view. We have reappraised some key research in this area and are now in the process of expanding the interface toolkit.

There are two interesting projects that gave some early inspiration and direction: the "Legible City" (Shaw, 1990) and "Osmose" (Davies, 1996) projects. Both developed a body-driven navigation interface for the particular non-realistic environments (a text analogised city in the case of Shaw and a surrealistic environment for Davies). In the case of the Legible City, the interface was an exercise bike and for Osmose, a scuba diving metaphor was used. In both cases, users could almost immediately concentrate on experiencing the VE rather than learning the navigation method, that is the interface very quickly became transparent to them.

Just as importantly, the sense of presence and connection with the VE was very strong. (Davies, 1998). The aim of the nAVRgate project is the same rapid transparency and sense of presence. In this sense the work of others such as Regenbrecht et.al. (2000) also aims to move the developments in this area in a direction that will produce more effective and appropriate systems for navigation and interaction.

It is, though, easy to fall into the trap of assuming that natural locomotion is best most appropriate for virtual environments. Thus far we have found it to be very effective, but are aware of the fact that assuming 'natural' is best may imply "questionable assumptions concerning distance and direction estimation and manoeuvrability" (Darken et.al. 1998).

In addition there are several pieces of work that show that spatial skills are not completely innate, and such findings have particular consequences when we consider designing virtual environments of architecture and for architects; which are not necessarily the same thing. We take up this issue, and the issue above later.

3. nAVRgate-1

It is clear that the method of interaction is a central part in the users immersive experience in a VE. Both the Legible City (Shaw, 1990) and Osmose (Davies, 1996) used a consciously non-realistic environment through which the experience was heightened by the use of a familiar real world navigational metaphor. The importance of retaining a sense of orientation in a VE has been noted (Bridges and Charitos, 1997). Furthermore, the problems of scaling and scalelessness in the locomotion through, and perception of, the virtual space require attention in developing such environments. It has been shown that proprioceptive information aids the human agent in relating motions of the body to movements in space. Loomis et. al. (1992) have shown that distance and direction estimation in virtual space were improved by the introduction of proprioceptive feed-back.

An aim of the nAVRgate project is the development of a generic navigational metaphor, that incorporate a variety of natural locomotion methods. We initially developed an architectural VE using a commercially available gaming engine and, initially, revisited the use of a bike for navigation.

nAVRgate-1 was ruthlessly pragmatic. An exercise bike (Knight and Brown, 1999) has been modified to give the basis of the system using a radically modified serial mouse. Left and right movement of the handlebars is translated into movement in the X axis, pedalling into the Y axis. Whilst this has overcome some of the problems some other researchers have found, such as the optical speed tracking of the wheel, it has raised others in the inflexibility of the scaling parameters of the standard mouse drivers. A purpose written driver would overcome this. The use of standard, modified components allows us to apply the system to other types of natural systems of locomotion such as walking (Slater et al, 1995) and using a wheelchair.

Back projection is used as opposed to a Head Mounted Display (HMD) for two reasons, firstly cost, but secondly, and more importantly, that it enabled group participation in the experience.

The software used was a standard games engine, Half-Life. Models were composed in a shareware level editor, WorldCraft and, whilst this was very effective for the task in hand, the limitation of the software soon became evident. As it is not possible to import CAD data directly into WorldCraft, models had to be (re)created from scratch. A further limitation was that, in common with other games of the time, it was not possible to model curved elements which resulted in either purely orthogonal models or crude misrepresentations. It was evident that a different piece of software would have to be used for the next stage.

User feedback has been very positive. The experience of using the bike has allowed us to learn a great deal about natural navigation metaphors. While users adapt very quickly to unnatural and in some cases slightly surreal situations (for instance, the remarkably entertaining experience of riding a bike down stairs to the sound of foot steps), a more natural locomotion and interaction device is still the goal. With the use in large scale urban environments in mind, nAVRgate-2 is a progression which will continue to use the exercise bike for distance (analogous with cycling between buildings), changing to foot travel using a treadmill or similar (analogous with walking around a building). However, rather than using buttons on the handlebars to change the view up or down, use of the body's natural movements is proposed.

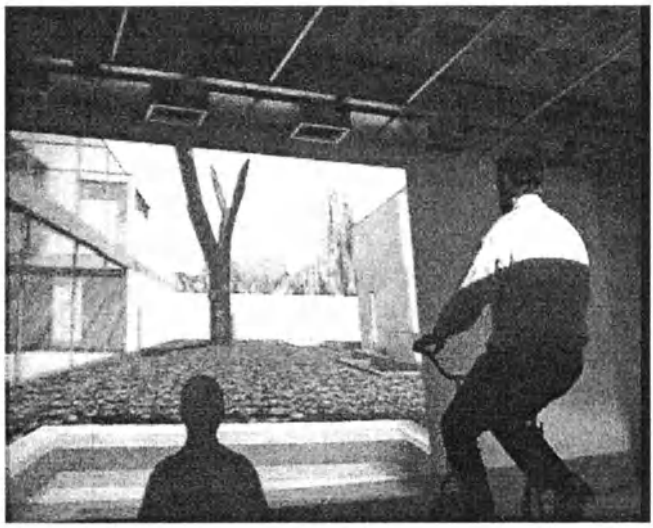


Figure 1. nAVRgate-1 bike in use

4. nAVRgate-2

When we experience a piece of architecture in reality, we look around using head movements. Whilst this is used in commercial products such as the 'Flock of Birds' motion tracking system which uses a signal source and a series of transponders to determine position, it does not fit well with our low cost aim. We aim to use the body's natural resources to provide the signals necessary for navigation.

The human head has three axes of movement, up/down (flexion and extension), left/right (rotation) and tilt. The muscles involved in the two that

we are particularly interested in, (flexion/extension for the Y-axis and tilt for the X-axis) are both large and near to the surface of the body. The sternocleidomastoid is present on each side of the neck; movement of the head to the left involves the right side muscle and visa versa. The trapezius muscle at the rear of the neck determines tilt. Muscles rarely work in isolation. For example, if both sternocleidomastoid muscles contract together, the head would tilt to the front, in this case the trapezoid muscle would be inactive. It is possible, through a series of logical conditions, by processing the combinations of electrical signals received, to determine the head movements.

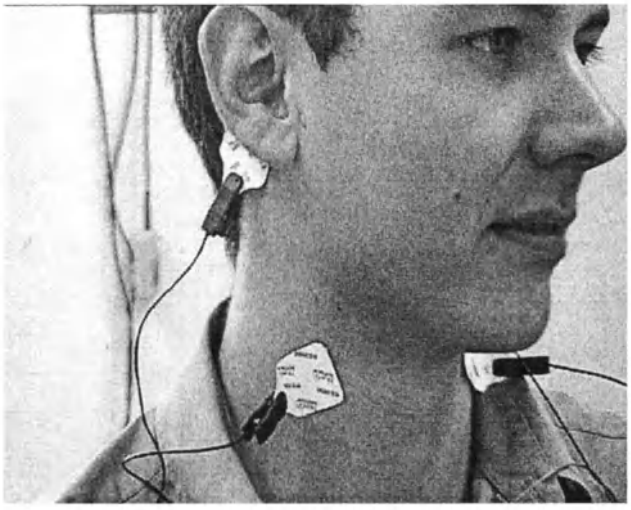


Figure 2. sensors in place

Whenever a muscle is activated a small electrical charge is generated, the size of which is dependent on the type of muscle movement. Muscles that control the head are in a state of constant activation to maintain posture. Movement of individual muscle fibres is evidenced by a small level of electrical activity, whereas in a large muscle movement involving all the muscle fibres, a much higher level of electrical activity is generated. This activity can be detected using surface fixed EMG (electromyography) sensors (Figure 2) similar to those used in detecting other bodily activity such as ECG heart monitoring. These are used as indicators of head movement. Once the threshold level has been determined, any deviation from this will be evidence of movement. For example, a movement of the head to the left would be indicated by a change in the signal strength of the right sternocleidomastoid muscle (Figure 3). In the extract of the signal

trace shown, both side of the neck show signal activity during the initial stages of movement due to changes in posture, but once in motion, the left side shows only threshold level of activity.

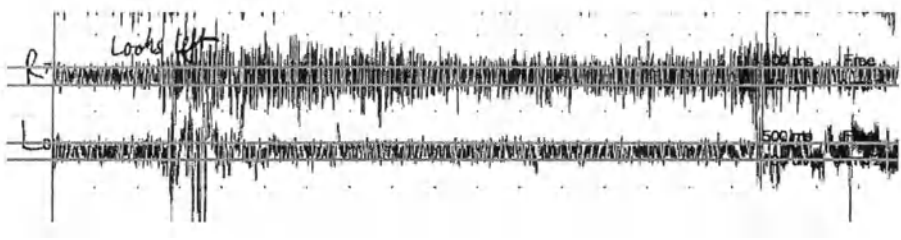


Figure 3. signal trace, head looking left

To date, we have established the feasibility of using the signals for detecting appropriate movement. The next stage will be the construction of a high-quality signal amplifier, interfaced to the system through an analogue to digital converter and processed to provide the necessary information to the locomotion governer.

The table below shows the muscle movements with the corresponding axial movements

Table 1. Logic matrix: muscle activation to head movement

		Sternocleidomastoid Left Side	Sternocleidomastoid Right Side	Trapezoid
-X	Left	-	**X**	**O**
+X	Right	**X**	-	**O**
+Y	Up	-	-	**X**
-Y	Down	**X**	**X**	-

Key: X *above signal threshold*
O *below signal threshold*
\- *static (no signal)*

As presented so far this technique sounds promising, but there are several problems with this approach. As with the rest of the body, muscles are several layers deep and there will be conflicting signals as each is triggered. Coupled with this is the variation in signal strength that varies slightly according to the size, age and build of the user. The software used to interprets the signals must be capable of being easily calibrated to compensate for this. A potential problem will arise when using a projected screen as the centre of gaze would shift with head movement, whereas the

projected image centre would remain static, problems which we do not see as insurmountable.

To continue to use the games software for the VE environments, the navigation needs to be compatible with conventional interface drivers. To use head movements, the software is configured to 'mouse follow' which means that the equivalent mouse X and Y movements correspond to the left/right/up/down movements of the view. Forward/backward movement is achieved with the keyboard arrow keys which are mapped to the appropriate interface (see system diagram, figure 4). This means that two axis of movement are required of the head, the X and Y, both positive and negative. A straight-ahead gaze would be 0,0, a look up would be a positive Y, a rotation of the head to the left would be a negative X.

To overcome the limitations of the WorldCraft/Half-Life gaming environment, we are currently experimenting with a games authoring package that will allow the direct import of CAD models in a variety of forms. This will, in turn, allow for a higher quality (and more accurate) VE to be constructed. The software also allows scripted events (sounds and behaviours etc) to assigned to individual objects which created an environment which can respond more readily to user interaction.

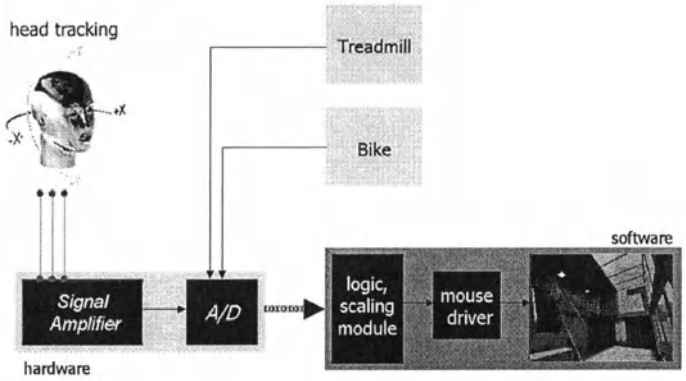


Figure 4. System diagram

5. REPRESENTATION OF ARCHITECTURAL SPACE

Control of locomotion, as a subset of the broader idea of navigation, in the Virtual Environment is one aspect of our work, but as we mentioned,

representation is another. This is an area that has particular implications for architecture.

Firstly we need to be aware of the particular spatial skills that aid us in navigation through three dimensional virtual environments such as those used in the nAVRgate system. Evidence shows that there is difference in spatial ability between individuals and genders (McGee, 1979), and this has broad implications. More specifically, research shows that skill in spatial manipulation and understanding can improve with practice and training (Carpenter and Just, 1986). Architecture is a very particular discipline; one where spatial abilities are developed and nurtured. Brown and Ryong (2000) have described and accounted for the particular cognitive skills that are developed throughout and architect's training; and the consequences, in terms of appropriate representations and interfaces are expounded.

Secondly, related to this work, and in particular to consideration of the architectural image Brown and Nahab (1996) have shown that the 'value' and perceived quality of an architectural scene is very dependent on the nature of the representation. Issues such as 'sketchiness', degree of photorealism and colour-monochrome rendering play a fundamental role in determining that perceived quality. In Virtual Environments, as with more conventional static representations, the nature of the rendered image (representation) surely plays a key role in generating an appropriate and successful ambience.

The issues mentioned above play an important part in our current thinking about how the nAVRgate system should be further enhanced. As we said at the outset the issues of VEs for architecture is a multifaceted problem, and appropriate representation is a crucial facet in that problem. As well as moving the interface issues forward, the next stages of the nAVRgate system will also address the nature of the appropriate image. Brown and Nahab addressed the static image, we intend to extend this work to include the images for VEs, particularly of the issues for large scale projection.

6. CONCLUSION

We have shown that it has been possible to implement and develop a low cost VE system for Architectural applications by taking ideas of natural locomotion and coupling that with developments spawned by games technology. The nAVRgate system that has evolved has proven effective and recent enhancements have been described here.

With the higher expectations of users today brought about by the ubiquitous application of computer generated animation, architectural VEs

must adapt and be flexible. It is a challenge for the world of architecture to keep pace with, and respond to these expectations. But we should do so from a particular standpoint. Architectural representation in three dimensions has a history that we can learn from and build on. Added to that, architects have particular needs and skills that developments, like the nAVRgate project, can respond to. Bill Mitchell (1995) talked about the idea of clicking "through cyberspace; this is the new architectural promenade". Our aim remains to elaborate on the click and to enrich the promenade.

7. REFERENCES

Bacelard, G, 1964 edition *Poetics of space* Orion, New York

Bridges, A. and Charitos, D, 1997 "The Architectural Design of Virtual Environments", *CAAD Futures 1997 München, Germany*, Kluwer Academic Publishers, pp. 719-732

Brown, A.G.P. and Lee, Hwa-Ryong 2000'A mental space model: towards computer augmented design', in *Greenwich 2000: Digital Creativity*, Greenwich, London, Jan 2000.

Brown, A.G.P. and Nahab, M. 1996 'Human interpretation of Computer Generated Architectural Images', in (Asanawicz and Jakimowicz eds) *CAD and Creativity,* Bialystok, Poland..

Bourdakis, Vassilis, 1997 "Making Sense of the City", *CAAD Futures 1997 [Conference Proceedings]* München (Germany), Kluwer Academic Publishers, pp. 663-678

Carpenter, P.A. and Just, M.A. 1986 1992, "Spatial Ability: An information processing approach to Psychometrics", *Advances in the Psychology of Human Intelligence,*, Vol.3, Hillabdale, NJ.

Cole, J. 1988 *Muscles in action: an approach to manual muscle testing,* Churchill Livingstone, Melbourne

Darken, R.P., Allard, T. and Achille, L.B., 1998, "Spatial Orientation and Wayfinding in Large Scale Virtual Spaces", *Presence: Teleoperators and Virtual Environments*, 7(2), p. 101-107.

Davies C., 1996 *The Paradox of Being in Immersive Virtual Space*, SIGGRAPH 96 panel, Soul in the Machine, The Search for Spirituality in Cyberspace

Davies C., 1998 *Changing Space, Virtual Reality as an Arena for Embodied being* in The Virtual dimension, Architecture, Representation and Cras h Culture ed Beckmann, J.

Eshaq, A.R.M. and Karboulonis, P. 2000, "The re-convergence of Art and Science: a vehicle for creativity", in: Tan, Tan and Wong (eds.) *CAADRIA 2000, Singapore May 2000,* pp 491-500

Gray, H. 1991 edition *Grays Anatomy* Courage Books

Knight, M and Brown, A.G.P., "Working in Virtual Environments through appropriate Physical Interfaces", in: Brown, Knight and Berridge (eds.) *eCAADe 17: From Turing to 2000, University of Liverpool, UK,* Sept. 1999, pp 431-436

Laurel B. 1993 *Computers as Theatre.* Addison-Wesley

Loomis, J.M., Da Silva, J.A. Fujita, N. and Fukusima, S.S. 1992, "Visual Space Perception and Visually Directed Action", *Journal of Experimental Psychology: Human Perception and Performance,* 18(4), pp. 906-921.

Maher, M.L., Simoff, S., Gu, N. and Lau, K.H.. 2000, "Designing Virtual Architecture", in: Tan, Tan and Wong (eds.) *CAADRIA 2000, Singapore May 2000,* pp 480-490

McGee, M.G. 1979, "Human Spatial Abilities: Psychometric Studies and Environmental. Genetic, Hormonal and Neurological Influences", *Psychological Bulletin,* 86(5), pp. 889-918.

Mitchell, W. 1995, *City of Bits,* MIT Press.

Remis, S.J, Neilson, D.K. 1989-90 *Force-Reflecting Interfaces to Telemanipulator Testing Systems Interim* Users report

Regenbrecht, H., Kruijff, E., Seichter, H. and Beetz, J. 2000, "VRAM: a virtual reality aided modeller", in: Donath (ed.) *eCAADe 2000: Promise and RealityWeimar, Germany, June 2000,* pp 235-238

Shaw, J. "Keeping fit (Mind and Body)", in: *Proc. @ HOME Conference: Doors to Perception 2, Amsterdam, Netherlands,* Netherlands Design Institute, Nov.1994

Winter, D 1979 *Biomechanics of Human Movement*, J. Wiley and Sons, Ontario

Virtual Environments for Special Needs

Changing the VR Paradigm

Tom Maver, Colin Harrison, Mike Grant
University of Strathclyde

Key words: virtual reality, mobility impairment, visual impairment, access, simulation

Abstract: The normal application of Virtual Reality is to the simulation of environments, which are in some way special - remote, hazardous or purely imaginary. This paper describes research and development work which changes the paradigm by simulating perfectly ordinary buildings for special people. Some 15% of the population have some form of physical impairment - a proportion which is likely to rise in line with an ageing population. New legislation, such as the UK Disability Discrimination Act places additional responsibility on building owners to ensure adequate access for people with an impairment and this in turn will place additional responsibility on the architect. Current methods of auditing access for new building are primitive and require the auditor to interpret plans/sections of the proposed building against a checklist of requirements specific to the special need. This paper reports on progress in the use of an immersive VR facility to simulate access to buildings for two broad classes of user: i) those with a mobility impairment; ii) those with visual impairment.

In the former case, a wheelchair motion platform has been designed which allows the wheelchair user to navigate the virtual building; a brake and motor connected to the rollers on which the wheelchair sits facilitate the effects of slope and surface resistance. In the latter case, the main categories and degrees of visual impairment can be simulated allowing architects to assess the contribution of form, colour and signage to safe access.

1. WHY VIRTUAL REALITY

There are many situations where it is more convenient, less costly, or indeed necessary to simulate our experience of three dimensional space rather than to experience, directly, its physical reality. The inaccessibility of

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 151–159.
© 2001 *Kluwer Academic Publishers.*

the real world may be as a result of its remoteness, its hazardous character or because it no longer exists, does not yet exist or indeed, never will exist:

1. remote environments:
   lunar homesteads, polar outstations, orbiting space stations and off shore oil drilling platforms are examples of spatial environments which are significantly remote; it is certain that convenience and economy can be served by providing an accessible simulation of the environment.
2. hazardous environments:
   environments contaminated by radioactivity, threatened by fire, structurally unstable or biologically inhospitable are either dangerous or expensive to make safe; economy and safety are promoted by appropriate simulation.
3. non-existent environments:
   these fall into three categories:
   - those environments which never will exist, e.g. the science fiction images of Hollywood movies.
   - those environments of architectural or historical interest which once existed but, due to the ravages of time are under threat or no longer exist, and
   - the hypothesised environments of architects and planners which it is intended, ultimately, to bring into existence.

An interesting example of the application of virtual reality to an environment which is remote, somewhat hazardous and partially non-existent in the Neolithic village of Skara Brae. Around 100 years ago on the coast of the Orkney Islands far off the north coast of Scotland, a severe storm uncovered part of a village believed to date from 2500 BC. It comprised a network of passageways and seven houses, complete with stone furniture, utensils and jewellery. Using digital video and Quick Time Virtual Reality, the Architecture and Building Aids Computer Unit, Strathclyde (ABACUS) created a virtual experience of part of the site (Figure 1), repopulating the houses with virtual representations of the artefacts and making conjectures regarding what was incomplete, e.g. the door opening/closing mechanism (Figure 2).

Quite clearly the benefits of such a virtual experience are widely available to the community. The benefits are particularly valuable to that proportion of the community who, by dint of physical impairment, would find that much more difficulty in experiencing the physical reality.

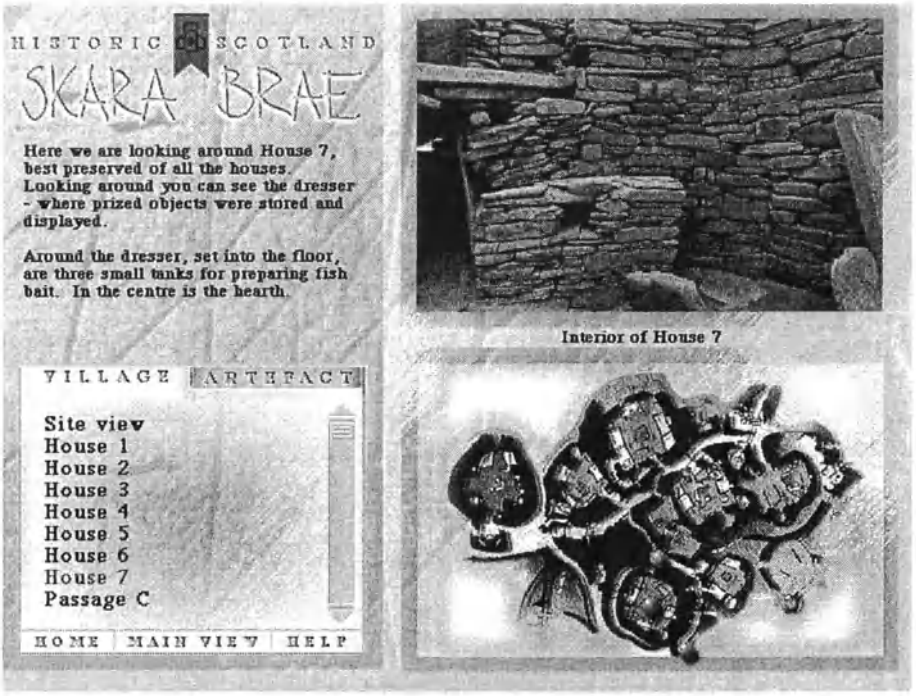


Figure 1. Home page of the Skara Brae Multimedia CD-Rom

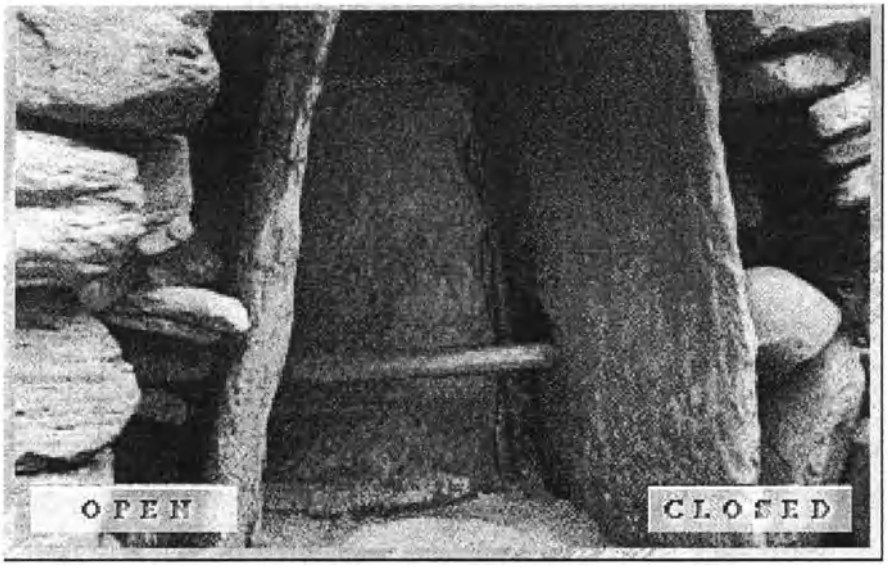


Figure 2. Interactive simulation of door-opening hypothesis

This paper is not, however, about the use of new technologies to facilitate virtual experiences of the built environment as it exists; much more importantly it is about the use of virtual reality of **change** existing and future built environments to enhance the experience of the real world for those with physical impairment.

2. SPECIAL NEEDS

Some 15% of the population have some form of physical impairment - a proportion which is likely to rise in line with an ageing population. New legislation, such as the UK Disability Discrimination Act places additional responsibility on building owners to ensure adequate access for people with an impairment and this in turn will place additional responsibility on the architect. Current methods of auditing access for new building are primitive and require the auditor to interpret plans/sections of the proposed building against a checklist of requirements specific to the special need. This paper reports on progress in the use of an immersive VR facility to simulate access to buildings for two broad classes of user:

a) those with a mobility impairment (approximately 3.5% of the population)
b) those with a visual impairment (approximately 6% of the population)

3. VIRTUAL ENVIRONMENT LABORATORY

ABACUS, in the Department of Architecture and Building Science at the University of Strathclyde is heavily involved in the use of virtual reality for a range of applications relating to the built environment and has a major facility, unique in Scotland, called the Virtual Environment Laboratory.

The virtual environment is visualised using a three-projector system that provides a 150-degree by 40-degree high resolution image mapped onto a five metre diameter cylindrical screen. Each of the three image channels is edge-blended to provide a seamless display. When viewed from any of the 14 seats in the Lab, the composite image fills the viewers' field of vision, providing a highly convincing sense of immersion within the scene.

The graphics images are generated on a 12-processor Silicon Graphics ONYX ll which is capable of processing detailed architectural models at high frame rates in order to provide realistic real-time animations. At each time-step in the simulation the graphics are rendered to three separate output channels, each channel sharing the same eye-point but with a different angular off-set in azimuth, corresponding to the off-sets in the projection

system. This circumvents the geometric distortion inherent in large field-of-view displays (Figure 4)

4. SIMULATION OF WHEELCHAIR ACCESS

ABACUS and the Strathclyde Bioengineering Unit are currently engaged on a research project funded by the Engineering and Physical Science Research Council to develop a virtual reality facility that can be used to generate, via an interaction between architects, designers and wheel chair users, guidelines which address the issue of wheelchair access to, and within, the built environment. The project aims to design and build a wheelchair motion platform through which wheelchair users can explore virtual representations of buildings. It is envisaged that such a facility would form a powerful and cost effective means of evaluating wheelchair access provision early in the design of new buildings and in the redevelopment of existing buildings. Accordingly the following preliminary objectives need to be met:

- The design and construction of a manual wheelchair motion platform that can accurately monitor intended wheelchair motion and can provide physical and optical feedback to the wheelchair user on the presence of virtual obstacles or changes in floor coverings or slope.
- Interface the platform with the Virtual Environment Laboratory facility to provide an immersive virtual environment within which navigation is linked to the intended wheelchair motion.
- Generate virtual representations of a range of building types in order to test and calibrate the performance of the platform and perform an evaluation of the system by wheelchair users.

The relationship of the wheelchair motion platform to the display system of the Virtual Environment Laboratory is shown in Figure 4.

The motion simulator and the graphics software are a close-coupled system. The motion simulator communicates with the control system is via a shared memory segment. The task of the motion simulator is to accept incoming data from the control system. This data relates to the individual incremental angular displacement of both wheels on the motion platform. This data is compared to the previous increment, to determine whether the wheel is rotated forward or backward, and to pass this information to the next stage of the algorithm. The basis of the motion control algorithm is the determination, through an analysis of similar triangles, of the location of the centre of rotation along the rear axle of the virtual wheelchair and the angle

Figure 3. The Virtual Environment Laboratory at Strathclyde University

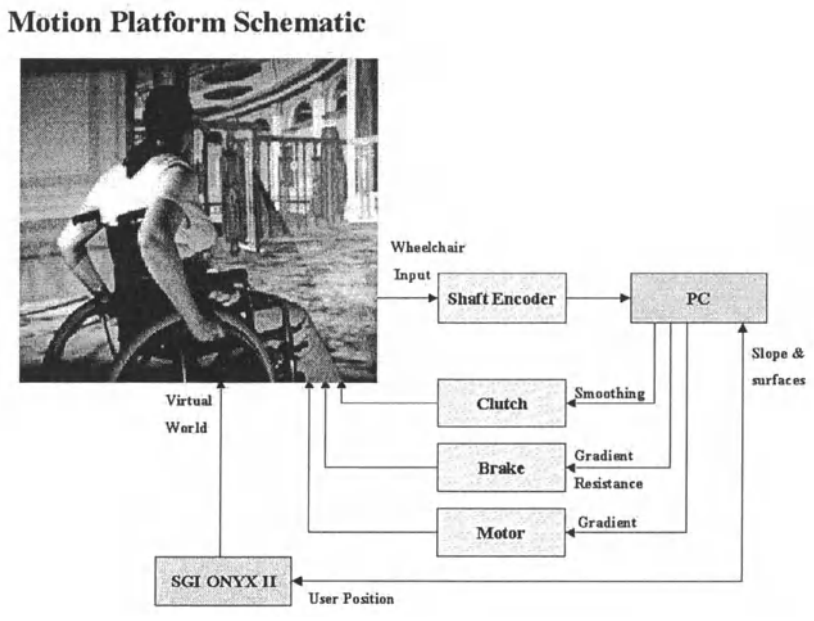


Figure 4.

through which is turned. From this the transformation of the eye point and rotation of the view vector can be determined.

The graphics application requires the Cartesian co-ordinates of the eye point, plus the yaw, pitch and roll angles of the direction of view. Given the yaw angle the remaining two parameters can be calculated based on the wheelchairs attitude on the floor plane. In the database traversal three rays corresponding to the contact patch of each of the rear wheels and the midpoint of the front axle, are intersected with the floor. The normal vector of the ground plane at these points can then be used to calculate the roll and pitch of the chair and the corresponding view. The same intersection procedure can also be used to identify the surface under each wheel, this information can then be used to index material properties, such as rolling resistance, which can be passed back to the control system.

The roller system is housed within a framework that supports the wheelchair and occupant, and converts wheel motion into an instrumented rotation of the main shaft. The system is duplicated for each wheel of the wheelchair. Mounted on each shaft are the brake, clutch, encoder, inertial mass and the take off for the motor drive.

The control system is based on a standard PC with purpose written software, that interfaces with the image generator. The control system feeds the motion engine of the image generator with incremental readings from the rotary encoders on the motion platform whilst controlling the feedback stimuli to the wheelchair on the basis of data received from the simulation in order to effect changes in floor conditions or collisions.

Each of the above elements forms a linked system that is controlled by the bidirectional flow of information from the wheelchair to the virtual environment, and from the virtual environment to the wheelchair. The feedback loop is by the users visual perception of progress through the virtual environment and by the perceived proprioceptive changes associated with alterations in the rolling resistance of the wheelchair. By closing the feedback loop with a human rather than a further pair of sensor connections, it is expected that any minimal latency or hysteresis in the rest of the communications path will be compensated for by the user. By April 2001 the wheelchair motion platform will be fully operational.

5. SIMULATION OF VISUAL IMPAIRMENT

It is currently extremely difficult to determine all the problems which someone with a categorisable visual impairment such as macular degeneration or glaucoma will experience when negotiating an ever-changing built environment. The current methods of assessing these

difficulties involve simulating the main categories of eye problems using either 2D fixed simulations (via modified slide transparencies) or adapted spectacles. Extended discussions with personnel from the Royal National Institute for the Blind and the Joint Mobility Unit – some of which took place following demonstrations within the Virtual Environment Laboratory– established that the current methods of simulating categories and degrees of visual impairment by using two-dimensional static simulations (in the form of modified slides or by the use of modified spectacles) are extremely limited. The slides cannot simulate dynamic movement through a space and neither slides nor spectacles offer a mechanism to appropriately simulate future, proposed environments.

There are five main causes of visual impairment: age related macular degeneration (48% of registrations); glaucoma (12%); retinopathy (3.4%); cataracts (3%); and retinitis pigmentosa (2%). Individually, or in combination, these eye conditions can affect perception of the visual field, visual acuity, stereopsis, dark adaptation and/ or colour vision. With the exception of stereopsis, the equipment in the VEL can support manipulation of these facets of the visual field through the virtual image. The project will focus on mapping these deficiencies, with the aim of establishing a methodology which can subsequently be applied to correlating forms of visual impairment and accessibility to spatial configuration.

The short-term objectives of the project are three-fold:

- To establish proof of concept and develop the means to model correlations between a range of dysfunctionalities caused by visual dysfunction and the actual effect this has on visual function when applied to accessing virtual and actual versions of the same environment.
- To establish whether the virtual reality environment can represent a real built environment, such that the experience of it maps sufficiently faithfully onto the experience of negotiating a real environment and can hence be correlated with actual forms of visual impairment.
- To identify and be able to virtually-represent the design features which need to be taken into account when designing an environment that optimises the variable abilities of visually-impaired people negotiating the built environment comfortably and safely.

By comparing the visual ability of respondents with and without various individual and combined visual impairments, also by undertaking these comparisons for both actual and simulated versions of the same route, it will be possible to establish whether the effect on realism and representation of visual impairment is greater than the effect of simulating buildings in the Virtual Environment Laboratory (VEL). This is an important issue for the simulation field, since if the simulation effect swamps participants visual ability then it will raise real questions about effectively simulating visual

impairment. If the effect is less, then this will confirm the scope to use virtual environments for developing access audits and other wayfinding analyses for the visually impaired.

6. IN CONCLUSION

The issues presented in this paper represent a shift in the paradigm away from the use of VR to simulate *special* buildings for *ordinary* people to the use of VR to simulate *ordinary* building for building for *special* people in the expectation that we can improve the design of existing and future buildings in order to enhance the experience of the built environment for that section of the user community with special needs.

The research and development required, however, is intellectually demanding and expensive; the authors are in no doubt, however, that appropriate investment in effort and resources will yield an outcome of great significance.

7. ACKNOWLEDGEMENTS

The authors wish to acknowledge, gratefully, financial support from the Engineering and Physical Sciences Research Council and the Scottish Higher Education Funding Council. They also acknowledge with thanks, input from colleagues in the Bioengineering Unit, the Royal National Institute for the Blind and the UK Joint Mobility Unit.

8. REFERENCES

Harrison, C., et al., "Development of a Wheelchair Virtual Reality Platform for Use in Evaluating Wheelchair Access', 3rd International Conference on Disability, Virtual Reality and Associated Technologies, Sardinia, September 2000, pp.1-8.

ABACUS, 'Virtual Reality Simulation Access to Buildings by People with a Visual Impairment", Grant Application to the Engineering and Physical Sciences Research Council, September 2000.

VR Sketchpad

Create Instant 3D Worlds by Sketching on a Transparent Window

Ellen Yi-Luen Do

Design Machine Group, Department of Architecture, University of Washington, Seattle, WA 98195-5720, USA

Key words: pen-based interface, freehand sketches, diagramming, transparent window, Virtual Reality Modelling Language (VRML)

Abstract: This paper describes VR Sketchpad, a pen-based computing environment for inputting and locating 3D objects in a virtual world. Designer can use the transparency layers to quickly trace and extract any image underlay from other application software. The 3D scene generation has three levels of complexity: simple extrusion of any drawn lines of shapes (i.e., straight or curved wall and column extrusion), solid modelling from a given geometric object representation (spheres, cones and boxes), and complex configuration with objects from graphics library (furniture layout).

1. INTRODUCTION

As on-line communication and entertainment enterprises increasingly use three-dimensional geometry, the ability to quickly create 3D content becomes important. Building three-dimensional worlds for the World Wide Web usually involves complicated CAD and modelling software that uses WIMP (windows, icons, menus, and pointers) interfaces. VR Sketchpad gives creators of three-dimensional content a quick and easy way to author 3D virtual worlds by sketching without coding VRML (Virtual Reality Modelling Language) or learning to model using CAD software.

With experience and training, designer can use CAD systems to create accurate models of 3D worlds. CAD systems are good at producing precise 3D models and supporting detailed editing and revision. However, pen-based systems enable designers to communicate ideas rapidly through approximate sketches with low overhead (direct manipulation by freehand drawing).

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 161–172.
© 2001 *Kluwer Academic Publishers.*

Sketching requires no precision or specialised knowledge, and makes it easy for designers to draw, evaluate, and explore as well as to revise and correct. Sketching is useful in settings that require a fast turn-around creation-feedback cycle, for example, during conceptual design. These settings include building and product design, as well as designing computer games and virtual worlds for VR applications, on-line exhibits, and cyber communities on the Web.

Providers of 3D content often start conceptual design by sketching with pencil or markers on paper or by building a physical model to simulate the proposed design. Later they use CAD software or game level editors to create these imagined 3D worlds. The process of making a 3D model with structured commands and operations is far more elaborate than using a pen and paper to sketch. A freehand drawing user interface for 3D modelling—as VR Sketchpad illustrates—provides designers more freedom to explore than a WIMP interface.

This paper presents VR Sketchpad, a freehand drawing interface for rapidly generating three-dimensional geometry in VRML format by tracing diagrams on a transparent window over an image source. VR Sketchpad employs a novel approach for creating 3D worlds. It enables designers to simply make diagrams for spatial partitions such as walls, columns and furniture on a 2D floor plan to quickly generate 3D worlds on the Web. For example, a quick sketch of a floor plan of an exhibition hall generates a virtual scene with walls and columns. Immediately, the designer can experience a virtual walkthrough from a Web browser. Preferred viewer positions marked on the floor plan are translated and embedded into the VRML viewpoint control panel, providing a guided tour path. As the visitor browses the virtual exhibit hall, VR Sketchpad displays the visitor's position on the floor plan sketch. This instant feedback from 3D VR world to 2D floor plan supports navigation and understanding of the created space for both visitors and designers.

2. RELATED WORK

Many projects investigate the use of transparent windows as an interface to facilitate information display or object generation. For example, PAD (Perlin and Fox 1993) uses transparent windows to support "zooming" from abstract information to more details. A reader can navigate through the document space by applying transparent pads on top of the area of interest. Toolglass and Magic Lenses (Bier, Stone et al. 1993) adopt a similar approach of having transparent objects carry with them functionality for user interaction. They employ transparency to allow users to construct new

artefacts with overlapping elements. Translucent Patches (Kramer 1994) use freeform shapes for copying and grouping information. The different patches can overlap and interact with each other (e.g., performing mathematical calculation of figures from overlapping patches).

Electronic Cocktail Napkin's trace layer allows selecting and copying of diagram sketches between different layers, it also supports re-arrangement of the layers through a post-up action with thumbnail representation of the drawings (Gross 1994; Gross 1996; Gross and Do 1996). The Napkin program also supports different pen types. Users of the system can leave marks in different thickness and darkness with the pressure and velocity data derived from the pen stylus. Trinder argues that architects use transparent media such as tracing paper and drafting film not only to bring images together, but also to compose configuration with hierarchy (Trinder 1999). His 'transparent medium' investigates using simple tools such as "push and pull" to translate pixel maps between two layers and a "sketching tool" that translate stroke information as different thickness. His empirical studies on twelve test subjects reveal that using transparent layer and sketching are a good way for design professionals to interact with computer software.

Several researchers also conducted usability studies of transparent user interfaces (Harrison 1996; Cox, Chugh et al. 1998). Their findings are encouraging. They found people could use transparent layers easily to shift their focus rapidly between the different views (e.g., a layer containing an instance of an element detail and an overview layer). This suggests that transparency user interfaces are useful.

Three-dimensional scene creation is of interest for many researchers at the human computer interaction paradigm and design research. For example, SKETCH is a system (Zeleznik, Herndon et al. 1996) that enables users to use gesture commands to specify the creation of 3D objects. Sketching three axial arrow lines will generate a box with corresponding dimensions. The efforts from Grimstead and Martin describe the method of constructing a solid model from a 2D line drawing. The process includes hidden line removal, line labelling, region recognition and alignment, and adjustment of vertices (Grimstead and Martin 1995). Digital Clay has a similar approach, however, unlike Grimstead's and Martin's approach, it uses freehand drawing as a front end interface instead of line drawing. It is a 3D isometric sketching environment that uses constraint propagation method of Huffman and Clowes (Schweikardt and Gross 1998) to infer the occluding, convex and concave edges to build 3D model.

Quick-sketch (Eggli, Bruderlin et al. 1995) automatically adjusts angles and connects freehand sketches to infer construction of geometric shapes and arcs in a 2D view. It also extrudes the plan profile to generate 3D shapes using object library from a graphical user interface toolkit. The project

DDDoolz allows user to add or delete connected voxel boxes in six directions (of the cube) to create 3D scenes (Achten, Vires et al. 2000). Teddy (Igarashi, Matsuoka et al. 1999) enables user to create spherical object models by drawing a 2D profile (a closed oval) then rotate the drawing sideways to draw the section for extrusion.

Our project, VR Sketchpad is well located in this series of research. It uses freehand sketching as a way to create 3D objects. On the most basic level, similar to Quick-sketch, VR Sketchpad uses extrusions to generate shapes. Beyond simple extrusion, VR Sketchpad also supports generations of solid objects such as boxes and spheres, as well as object placements such as furniture layout creation from 2D symbol configurations.

3. VR SKETCHPAD IN ACTION

VR Sketchpad employed Electronic Cocktail Napkin's various capabilities of diagram parsing and customised user training (Gross 1994; Gross 1996; Gross and Do 1996) as a base graphic engine. The symbolic processor in the Napkin program enables designers to train and define their own graphic symbols and gestures (circle, line, arrow, etc) by drawing examples into the system (with a pen and tablet). Contrary to the approach of PDA (Personal Digital Assistant such as Palm Pilot) that can recognise only a fixed set of shapes, our system lets designers create their symbol library with personalised definitions and drawing features (sequence, pressure, angle, etc). VR Sketchpad then provides meaning association (circle to column, arrow to viewpoint) and performs geometry translations. Figure 1 below illustrates the system architecture of VR Sketchpad.

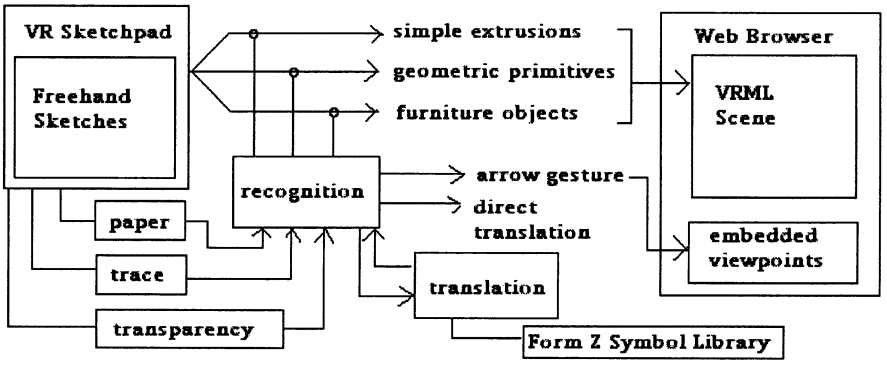


Figure 1. System architecture of VR Sketchpad

Three types of interfaces are available for the sketching input. First, the regular drawing board, paper like interface that allows designers to sketch onto a digitising tablet using a pen stylus. The drawing surface is opaque and displays sketches as drawn. The second type of interaction allows bringing in a picture underlay (raster image) and multiple translucent layers on top. Figure 2 shows a construction drawing brought in to the paper drawing board environment. Several trace layers are overlaid on top to add modifications and annotation diagrams. The thumbnail images on the top portion of the drawing board ("post-it" layer) allows easy management (hide, overlay) of the different trace layers. Each trace layer adds opaqueness to the drawing board while underlay drawings remain visible through the layers. Figure 3 shows a hand drawn diagram on the transparent window overlays on top of other applications (e.g., Form•Z model, screen capture of a drafting document, and a PDF file). The transparent window maintains the drawing functionality of the previous two types of interface (paper, trace layer).

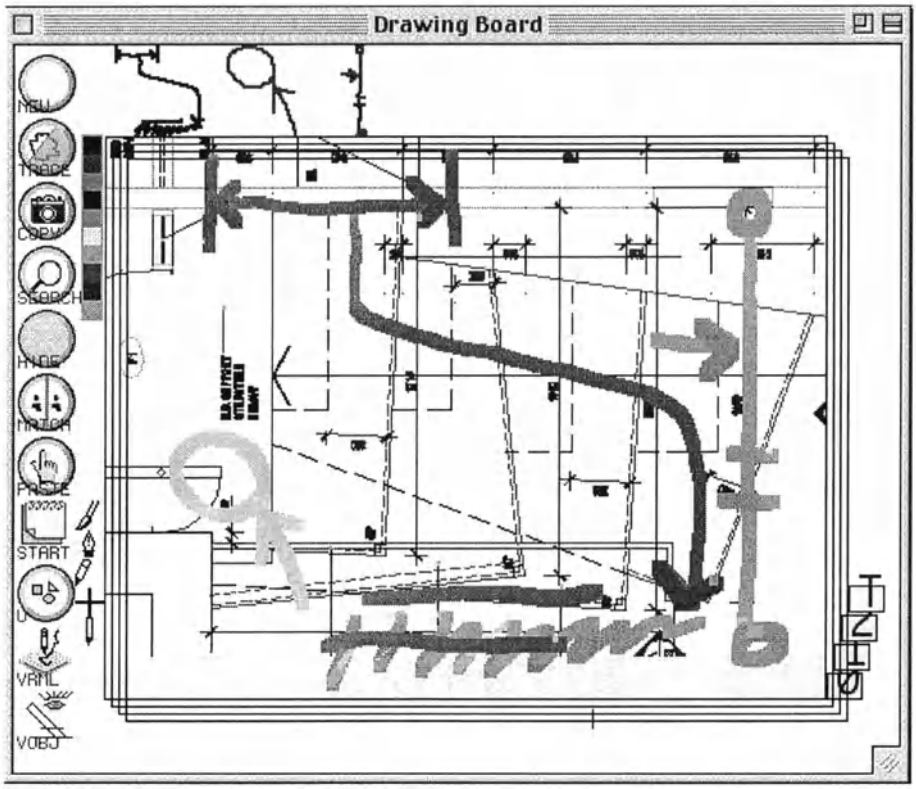


Figure 2. A construction drawing brought in as picture underlay with annotations and modifications on different trace layers. (lower right corner shows tabs for each trace layer for easy selection).

A transparent window is a window (drawing surface) that displays the screen's pixel map as a background for its contents. It appears as though you are drawing on top of the windows belonging to other software applications, or the operating system. With inter-application communication protocols, the Sketchpad would be able to directly interact with the underlay applications. Currently, when user resizes or moves the window, the transparent window instance captures the screen pixel map underneath, and builds the bitmap information into a memory cache. The content drawing method of the window draws the background images first then displays the current user sketch objects.

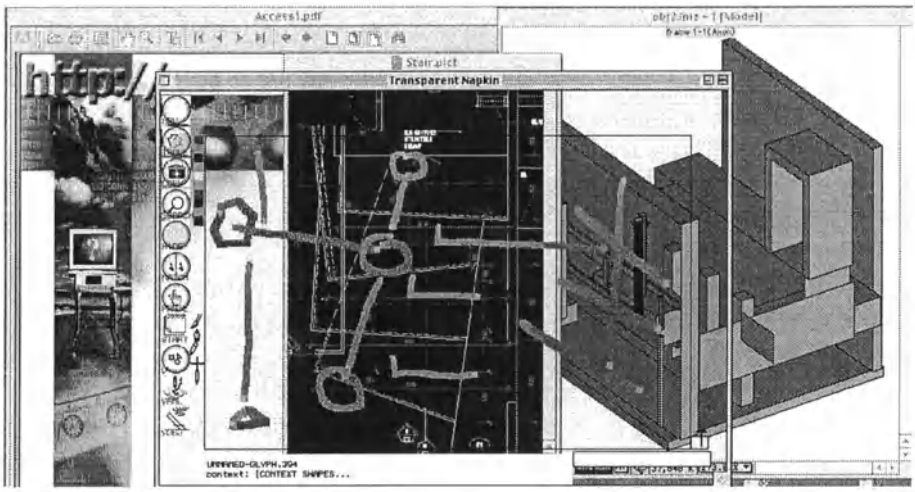


Figure 3. Transparent window overlays on top of three different applications (left to right: PDF document, drafting document, and a Form•Z model). User can sketch and trace images at the transparent window.

With initial input of sketches, VR Sketchpad's processor recognises the drawings (Figure 4 left), it translates the drawing into VRML objects, and then launches a Web browser to display the result (Figure 4 right, VRML enabled Netscape browser).

VR Sketchpad has three levels of diagram recognition and translation processing. First, simple shapes such as lines and circles are extruded to make walls and columns as shown in Figure 4.

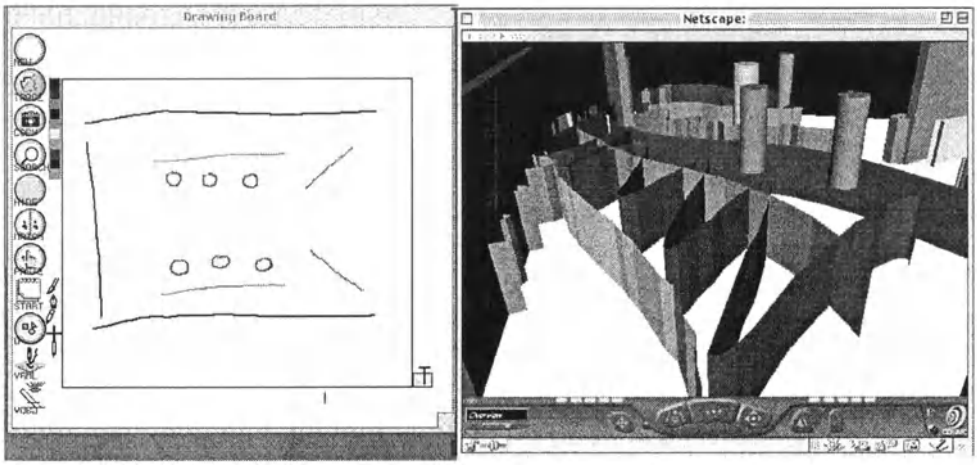


Figure 4. Lines and circles are extruded to make VRML model of walls and columns.

The basic level of translation is simple extrusion. VR Sketchpad also supports extrusions of free form, irregular shapes as desk height (wall height is to the ceiling). The translation processor of VR Sketchpad takes all the user input points in the stroke and converts them into VRML surfaces.

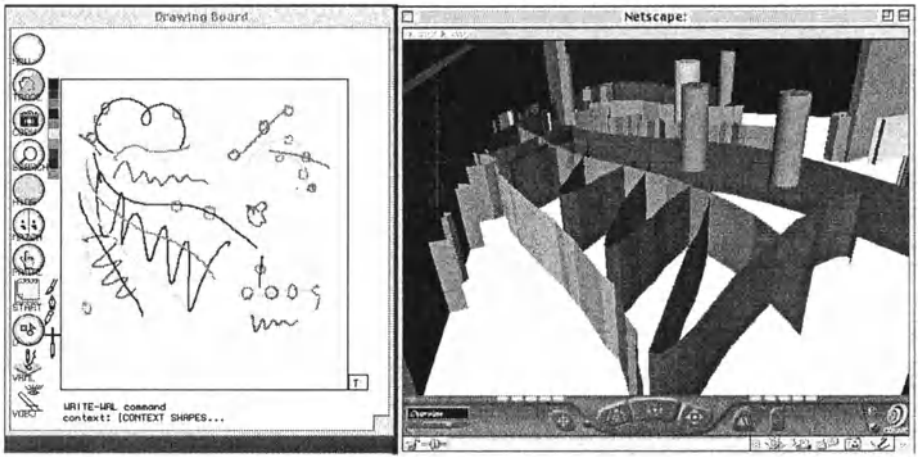


Figure 5. All curve shapes and lines can also be extruded to make curvy partitions. (Note: in this illustration, cylinders are inserted over the partition walls).

The second level of diagram processing deals with solid object mapping and placements. Instead of extrusion, it recognises geometric primitives through symbol conventions or configurations. For example, a rectangle is identified as a box object and translated as a solid cube. A circle that is concentric with a dot indicates a sphere object with desired radius. Figure 6 shows a VRML scene constructed from sketches of geometric primitives.

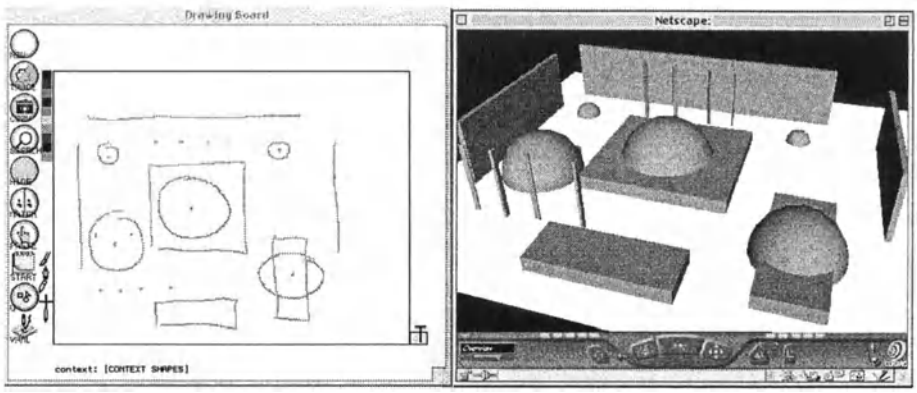


Figure 6. Graphic symbols can be translated into solid VRML object models.

The third level of processing involves recognition of the diagram configurations and the translation with calling of external 3D pre-made objects from a symbol library (e.g., furniture library from Form•Z). Figure 7 shows some shorthand symbols of architecture elements for kitchen and bathroom. Designers can train the configuration processor to recognise graphic standards or personal symbols by drawing freehand diagrams. For example, a toilet can be defined as a rectangle directly above two concentric circles; a dining table set can consist of four chairs (rectangles) surrounding a round table (circle) as shown in Figure 7-9.

Figure 7. Graphic symbols for stove top, sinks, toilet and bath rub.

With the combination of "level one's" partitions (walls and columns) and the translations of higher level complex figures (furniture elements from 3D symbol library), VR Sketchpad user can quickly sketch out an interior layout configuration and see the design in 3D VRML format by pressing a button. Figure 8 shows a furniture layout diagram and the corresponding 3D scene. Notice that all the furniture objects are in real human scale because we use real life object symbols from the 3D graphic library. Diagrams in this instance serve as placement reference instead of scale.

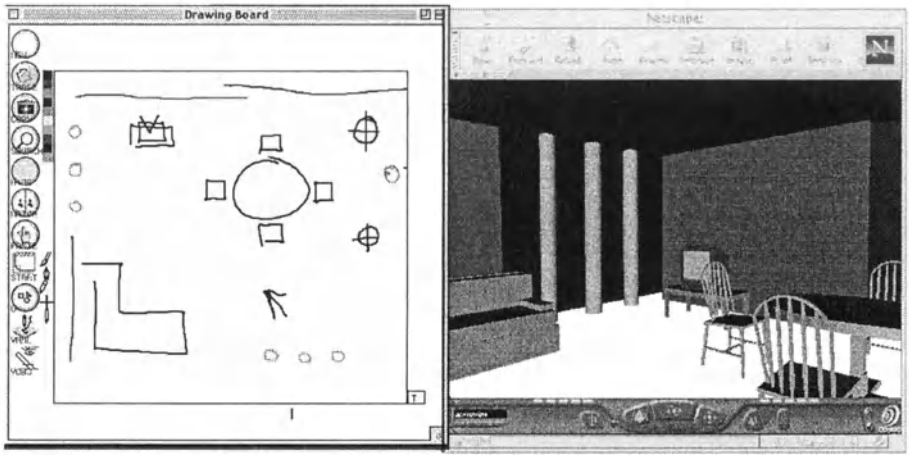


Figure 8. Furniture layout sketch (TV, couch, dining table set, columns, and walls) creates a 3D VRML world.

One important feature of VR Sketchpad is that it also supports gesture recognition for user to specify preferred viewpoints in the 3D VRML scene. The arrow in Figure 8 defines a standpoint and viewing direction toward the scene. The Web browser window on the right shows a particular view angle as indicated on the drawing on the left.

Figure 9 shows that user can draw a sequence of arrows to indicate location of interests and therefore define a viewing path into the 3D world. There is no limit on how many kinds of object configurations or how many objects can be placed in a 3D VRML scene (memory permitting). Users can define their own shorthand symbol sketches and use them to indicate desired element placements. The 3D scene in Figure 9 shows a particular view (lower left arrow on the floor plan) behind the columns looking toward the dining table and chairs, with walls and columns in the background.

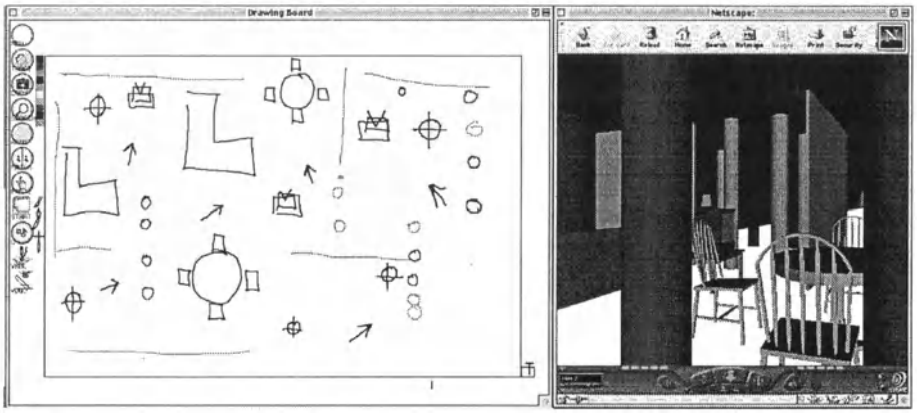


Figure 9. A sequence of arrows in the floor plan sketch indicates places of interest and provides a guided tour to the 3D VRML scene. The bottom left arrow on the drawing (left) shows a location and view into VRML scene on the Web browser (on the right).

4. DISCUSSION AND FUTURE WORK

VR Sketchpad is an application that allows user to sketch in two-dimensional freehand diagrams to quickly generate three-dimensional VRML scenes on the web. The idea of the project came from the teaching of beginning architecture students. The novice architecture students have a hard time to see the relations between the two-dimensional floor plan as a three-dimensional space. The obvious tool for the students to understand the interplay between plan view and its corresponding space is to extrude their 2D drawings into 3D form. Therefore, VR Sketchpad was created to illustrate the relationship between 2D and 3D. VR Sketchpad has then added many features and now supports not just direct one-to-one mapping of object translations but also the higher level complex figure configurations and substitutions (diagram meaning, as well as replacement object symbols). It also shows great potential for on-line gaming and artefact generation for the Web.

The system implementation is straightforward. The interface is simple, yet powerful. Many designers requested to test the system once they saw it in action. We plan to release the prototype VR Sketchpad to architecture students to use in their design studio for conceptual design. We would also like to conduct formal usability studies of the system to find out the problems and desirable functionality.

We are currently adding a slider function in the drawing palette to allow user to specify the extrusion heights. We are exploring adding a 2D sectional view with the 2D floor plan view. We are extending VR Sketchpad to support creating virtual terrain with freehand sketches. Another direction of

the project is to provide an interface to support the display and selection of product catalogue (e.g., Sweets and Steelcase) items with hand drawn symbols (wall, furniture, etc.). We would like to add VR Sketchpad as an interface to a modelling program (e.g., autodessys inc. recently announced its plan for providing interface into their Form•Z software). We have chosen to implement in VRML format because of its ISO (International Standards Organisation) status and because all modelling programs have translation modules to read it as input.

There are many questions worth investigating. Should we extend 2D sketch with isometric view extractions? A previous project Digital Clay (Schweikardt and Gross 1998) from our research group addresses exactly this concern. What does it mean to be able to sketch in 3D? Shall we sketch into the VRML scene to modify the design "right on the spot"? A project in our research group called Space Pen (Jung 2001) is currently exploring this aspect of sketching interactions in 3D.

VR Sketchpad works, but it also has flaws. For example, the furniture placement function uses only the sketch locations as reference for placement and discards the dimensional info (because the furniture models are constrained to real dimension). We have explored placing element dimensions according to user's sketches. However, this would generate scaled down (or up) furniture (that's not usable in real life). It seems to be a logical next step to embed constraints into the sketches such as Stretch-a-Sketch (Gross 1994) so that user, while drawing, will be aware of the human scale concerns and ergonomic dimensions. Or one could display scale and a grid underlay in the drawing surface. We also plan to extend the transparent windows as an interface to other applications. For example, sketching over a VRML scene on a Web browser could potentially add new elements to the 3D scene.

5. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant numbers IIS-96-19856 and IIS-0096138. The views contained in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

6. REFERENCES

Achten, H., B. D. Vires, et al., 2000. *DDDOOLZ.* CAADRIA 2000. B.-K. Tan, M. Tan and Y.-C. Wong. Singapore, National University of Singapore: 451-460.

Bier, E. A., M. C. Stone, et al., 1993, "Toolglass and Magic Lenses: The See-Through Interface", *Computer Graphics* (SIGGRAPH '93 conference proceedings), 73-80.

Cox, D. A., J. S. Chugh, et al., 1998. *The Usability of Transparent Overview Layers.* CHI 98: 301-302.

Eggli, L., B. D. Bruderlin, et al., 1995. *Sketching as a Solid Modeling Tool.* Third Symposium on Solid Modeling and Applications. C. Hoffmann and J. Rossignac. Salt Lake City, ACM: 313-321.

Grimstead, I. J. and R. R. Martin, 1995. *Creating Solid Models from Single 2D Sketches.* Third Symposium on Solid Modeling and Applications. C. Hoffmann and J. Rossignac. Salt Lake City, ACM: 323-337.

Gross, M. D., 1994. *The Fat Pencil, the Cocktail Napkin, and the Slide Library.* ACADIA '94. M. Fraser and A. Harfmann. St. Louis, MO: 103-113.

Gross, M. D., 1994. *Stretch-A-Sketch, a Dynamic Diagrammer.* Proceedings of the IEEE Symposium on Visual Languages '94. A. Ambler, IEEE Press: 232-238.

Gross, M. D., 1996, "The Electronic Cocktail Napkin - working with diagrams", *Design Studies* 17 (1), 53-69.

Gross, M. D. and E. Y.-L. Do, 1996. *Demonstrating the Electronic Cocktail Napkin: a paper-like interface for early design.* CHI 96, Conference on Human Factors in Computing Systems. Vancouver, British Columbia, Canada, ACM. Conference Companion: 5-6.

Harrison, B., 1996, *Design and Evaluation of Transparent User Interfaces.* PhD: Toronto, University of Toronto.

Igarashi, T., S. Matsuoka, et al., 1999, "Teddy: a sketching interface for 3D freeform design", *Proceedings of the SIGGRAPH 1999 annual conference on Computer graphics* , 409-416.

Jung, T., 2001. *Space Pen: annotation and sketching on 3D models on the Internet.* CAAD Futures 2001. Eindhoven, Kluwer Academic Publishers.

Kramer, A., 1994. *Translucent Patches - dissolving windows.* ACM Symposium on User Interface Software and Technology, Marina del Rey, CA, ACM Press.

Perlin, k. and D. Fox, 1993. *PAD: an alternative approach to the computer interface.* SIGGRAPH. Anahelm: 57-62.

Schweikardt, E. and M. D. Gross, 1998. *Digital Clay: Deriving Digital Models from Freehand Sketches.* Digital Design Studios: Do Computers Make A Difference? ACADIA 98. T. Seebohm and S. V. Wyk. Quebec City, Canada, ACADIA: Association for Computer-Aided Design in Architecture: 202-211.

Trinder, M., 1999. *The Computer's Role in Sketch Design: A Transparent Sketching Medium.* Computers in Building; Proceedings of the CAAD Futures '99 Conference. G. Augenbroe and C. Eastman, Kluwer Academic Publishers: 227-244.

Zeleznik, R. C., K. P. Herndon, et al., 1996, "Sketch: An Interface for Sketching 3D Scenes", *SIGGRAPH '96* , 163-170.

[roomz]&[connectionz]

Scenarios in Space and Time

Kai Strehlke, Maia Engeli
Swiss Federal Institute of Technology
Zurich, Switzerland

Key words: Virtual Reality, Three-dimensional user interface, Narrative structures, Web-based Environment, Digital space design, Architectonic representation.

Abstract: New opportunities for architectonic design and representation in three-dimensional virtual spaces have arisen thanks to computers, networks, and new media in general. An environment with unique three-dimensional interfaces for the creation of spatial scenarios has been developed. The possibilities were explored in two successive courses in architecture and new media. Design in space and time, the investigation of special characteristics of digital spaces and the creation of a hyperstructure were the main topics of the creative work.

1. INTRODUCTION: EXPLORING DIGITAL SPACE

The formulation of architectonic ideas through digital three-dimensional spaces is gaining importance in the architect's work, from designing purely virtual architecture, like computer games or information spaces, to presenting planned buildings using virtual reality. Above and beyond this area's many practical applications, our interest and the focus of this paper lies in exploring the potential of these new possibilities, defining appropriate tools and interfaces, and training architects to use new media to reveal their ideas.

To be able to design three-dimensional spaces in an immersed way has become a desire among architects working with CAD tools. The challenge is to create tools that support the designer's creative thinking, allow for intuitive interaction, stimulate the design process, and help one discover additional qualities of the spatial composition. In a virtual environment, the

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 173–186.
© 2001 *Kluwer Academic Publishers.*

special qualities of digital spaces and their differences to physical spaces have to be considered in order to discover their full potential. The lack of gravity, orientation, and physical scale on the one hand, open opportunities by creating freedom for expressing ideas and, on the other hand, create challenges because the audience is not yet trained to cope with all of the aspects of this freedom. "In a participatory medium, immersion implies learning to swim" (Murray, 1997).

The design as well as perception of space goes beyond geometry and materialization. Context, event, mood, and memories influence the experience, which is always personal and subjective. The same space can have a multitude of parallel interpretations resulting from the fact that different authors are involved. To communicate such complex interrelated aspects is a challenge that can be addressed by using networked environments as a medium for representation and communication. The network serves as context and allows parallel presentations of the same space. The messages should be thought of as stories that stimulate imaginative thoughts among the audience.

The tools and interfaces created for [roomz] specifically address these aspects of revealing architectonic ideas within three-dimensional digital spaces through narrative scenarios. In [connectionz] these scenarios can be networked into a hyperstructure to allow for a multithreaded, multifaceted representation created by one or multiple authors.

"Writing stories is creating knowledge and the sharing of stories defines a culture" (Schank, 1995). The vision is that the scenarios of [roomz] become architectonic stories, and the hyperstructure of [connectionz] becomes a multithreaded narrative - a collaboratively-created architectonic knowledge space that exemplifies what Rushkoff describes as "a much more self-conscious, recapitulated experience of storytelling" (Rushkoff, 1996).

2. DIGITAL ENVIRONMENT AND INTERFACES

The environment for the creation of spatial scenarios is implemented as a set of web-based interfaces. The goal of these interfaces was to allow an author to design directly in a three-dimensional space in a very intuitive and interactive way. The editing environment allows one to place visual material in the form of images, video clips or three-dimensional objects in a given geometry and to define a path through this geometry. The scenario results from combining the carefully placed objects with the motion along the path. Ultimately, a scenario is about events, which occur within a space. "Events qualify spaces as much as spaces qualify events" (Tschumi, 1996). This

contingent relationship can be applied to and explored in [roomz]&[connectionz].

The scenarios are about digital spaces, which are not bound to the physical world and do not have to be a simulation of it. A scenario can create a space of information, a space of memories or feelings. In addition to the objects, the path is an important part of the scenario. The path allows the author to guide the reader's perception of the space.

In addition to the interfaces provided by [roomz], which allow the creation of single scenarios, the environment of [connectionz] allows one to interconnect scenarios and create a multidimensional hyperstructure. Hyperdocuments, as known from the web, are a challenging way to provide information, challenging both for the authors as well as the readers. A hyperdocument is comprised of rich multidimensional information structures where readers can enjoy travelling along various paths (Engeli, 2000). The same paradigm is applied in [connectionz] when connecting information to architectural spaces allows the architectural discourse to happen within one scenario as well as in relation to other scenarios.

2.1 Editing in Space

The most direct approach for visualizing architectural ideas about space is by directly working in three-dimensional models. Therefore a three-dimensional workspace has to be provided that allows the designer to work in a very direct way and immediately experience the result of any design decision.

The main interface of the [roomz] workspace is called "myscenario". This interface allows for three types of interactions: changing the colors of the walls, placing objects within the space, and creating a path through the space.

Myscenario consists of the three-dimensional space the author is working on and a two-dimensional part providing the tools to create the scenario. The tools are implemented as a head-up display, which is perceived as an overlay over the three-dimensional space. This overlay technique has been successfully implemented in previously developed design tools like Sculptor (Kurmann, 1998) and xWorlds (Strehlke, 1999), leading to an enlarged modeling space and an enhanced involvement with the task at hand when compared with the traditional setup of menus and tools arranged around the workspace.

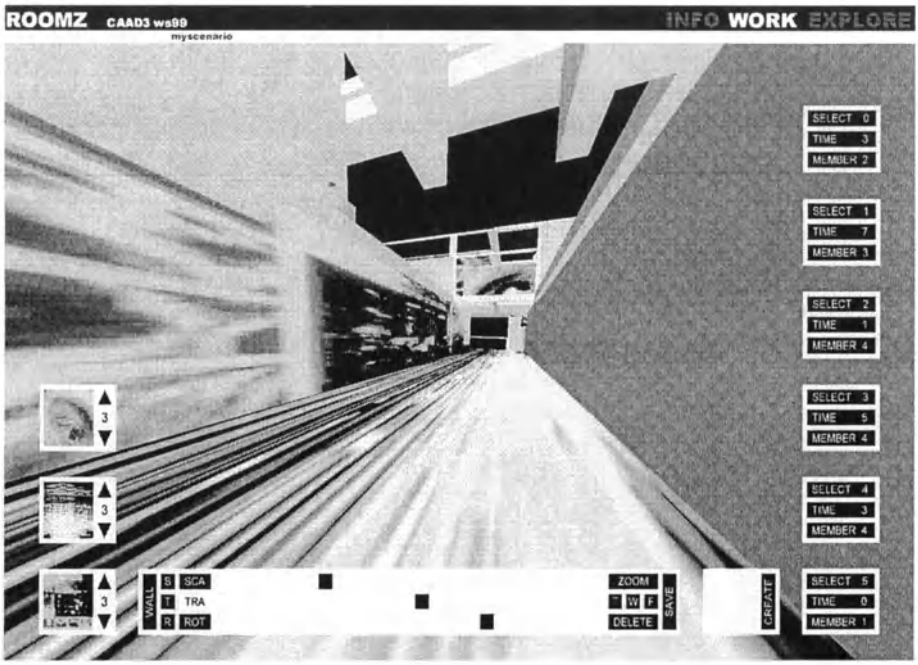


Figure 1. Interface for placing visual objects into the spaces and generating a path through the scenario

Before working in the main interface, the author has to import the visual objects that should be placed in the spaces. A special interface is provided for this preparatory task. Three types of objects can be imported: images, video clips and three-dimensional objects.

The head-up display is divided in three parts, showing the imported objects on the left, the viewpoints of the path on the right and a control panel on the bottom. The panel contains sliders to control the properties of various aspects of the objects and the path.

2.2 Visual Objects and Instances

"The skins and surface envelope of buildings become programmable surfaces, photosensitive membranes that narrate, design and inform the spatial organization of the volumes and interpret their functions. Information loaded walls that seduce" (Colafranceschi, 1995).

Visual objects provide the means to formulate the scenarios. Because the objects are always present in the workspace, the designer is able to work with them like a painter is working with a palette of paints in his hand. Multiple instances of an object can be placed in the scenario. One only has to click on the icon to create a new instance. The instance can be placed freely

in space or pasted on a wall of the model. Images can be pasted with any proportion and size by clicking on the first corner of the instance and dragging the mouse to the opposite corner. Using the control panel, images can be stretched in both x and y-directions, moved in any direction, and rotated. Furthermore, it is possible to scale the image within the instance, resulting in a repetition of the image on the same plane.

Different possibilities are enabled depending on the instance. When an instance is placed freely in the space, it is possible to freely rotate it. In a situation where this does not make sense, like when the image is pasted on a wall, free rotation is disabled. Such features enhance the performance of the designer as well as the tools by reducing meaningless efforts, and are available for video clips as well as for images. Different instances of video clips can also be created and either pasted on a wall or placed in space. The videos can be activated when they are clicked upon.

The third group of visual objects is made up of three-dimensional models imported as VRML models. Their instances are placed in space and, if necessary, the location and scale can be readjusted with the controls. The VRML models allow the space to be filled with dynamic objects.

In the physical world, a reference object is always needed in order to be able to perceive the size of a space. However in the virtual world, it is possible to redefine spaces with dynamic, moving objects, which can also use sound to enhance the scene.

2.3 Creating a Path

"Motion helps us gather information not available at a glance. It moves us from one space to another absorbing data and assembling the spaces mentally from various brief glances. Our cognitive model, while schematic, is in a way more complete than a static perception of the space" (Anders, 1998).

Motion is essential for understanding the third dimension in a digital space, since the computer screen can only display 2D images. By looking at objects or spaces from different angles, it is possible to read and understand the three-dimensional geometry.

Navigating in a virtual three-dimensional space is a new challenge when compared to navigation in the physical world. The virtual space may lack gravity, scale and horizon. One may not perceive it from a constant height and the viewing angle can change while moving. To recognize a larger context or find landmarks for orientation may be very difficult, especially when jumping from one viewpoint to another; one can easily get disoriented and eventually end up on an empty black screen.

Therefore, navigation is an important part of the whole scenario. Since the content is deliberately placed in the space, it is also important for the author to help the visitor perceive the space in the way he or she wants him to see it. Every viewpoint, camera opening angle, the speed when traveling from one viewpoint to the other, and the elements which are visible at a certain time are crucial aspects that can be controlled. The author can freely navigate in his working environment when designing the scenario, but the visitor is constrained to follow a designated path when exploring it.

Half of the myscenario interface is devoted to the design of the path through the scenario. The path consists of a sequence of viewpoints. For each viewpoint, different parameters can be defined, like the location and orientation of the viewpoint, the viewing angle, and the time needed to reach the viewpoint. The lights in the model can be adjusted and whether or not each instance of a visual object is visible or not can be determined for each viewpoint.

These possibilities allow different approaches for formulating the path. It is possible to follow the paradigm of moving through a physical world by simulating a horizon, keeping a defined viewing angle and viewpoint height, and by moving in a more or less constant speed from one viewpoint to another. Furthermore, it is possible to tilt the space from one viewpoint to another, to change the viewing angle, or to alter the speed, from fast jumps to very slow motion, between the viewpoints. Instances can also appear, disappear or be replaced when a viewpoint is reached. With this rich palette of possibilities, how one moves through the space becomes a non-trivial design task.

2.4 Creating the Hyperstructure

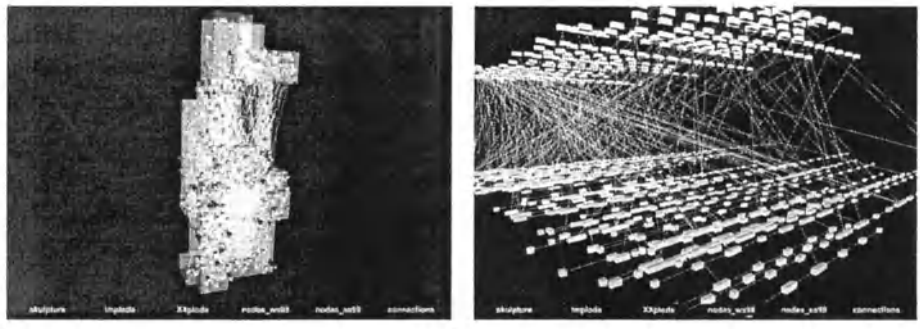


Figure 2. Visualization of the hyperstructure created by all the scenarios from the two courses [roomz] and [connectionz]

The environment of [connectionz] allows scenarios to be linked together into a hyperstructure. The links are created by introducing gates which lead

from one scenario to another. By clicking on a gate, the viewer jumps for a defined number of viewpoints into the other scenario before being brought back to the first scenario. This allows moving through the spaces in various ways and ultimately gives access to all connected scenarios.

Links, as we know them from hyperdocuments, generally lack transparency or other means to anticipate where they lead. However in the [connectionz] environment, supporting the user's decision to follow or not to follow a link seemed important to us. Two means of enhancing the transparency of gates were therefore introduced: Gates needed to be decorated with an object from the linked scenario, and clicking on the gate for the first time switched on a preview of the linked scenario. A double click let the visitor move into the linked scenario.

An important aspect to make a visit in the [connectionz] environment enjoyable is to prevent broken links. Therefore the following measure was taken: If the author of the foreign scenario erases the part of a path a gate points to, this gate becomes meaningless and is removed by the system. Even though the visitor is spared some frustration, the scenario's author may gain some frustration when a gate is lost. Connections to old and finished scenarios are stable, while connections to scenarios under construction can be unstable. Connecting to an unfinished scenario means that the author does not have full control of all aspects of his work, which is more and more a fact when working in networked environments.

2.5 The Technical Setup

The interfaces run in typical web-browers with standard plugins on the client side and are implemented with shareware tools on the server side. This approach minimizes the user's effort to become acquainted with the system and provides the possibility to work from anyplace anytime, because the interfaces are system independent and require no special software. On the server side, a MYSQL database and a set of PHP scripts create dynamic and personal HTML and VRML interfaces for the user. On the client side, a Netscape browser with a VRML plugin is needed for the author to work on his or her scenario.

Two-dimensional HTML interfaces are provided for authentication, uploading material into the system, and retrieving information like tutorials. Three-dimensional VRML interfaces, like the myscenario interface, the gates interface, and the explore interface, allow one to work in the spaces and explore them.

In the case of the myscenario interface, a PHP script generates a personal VRML world, which contains the geometry of the user's space, the objects already uploaded into the database, the instances, the path the user has

created, as well as all the tools needed for modifying them. When the work is saved, a connection to the server is opened and all of the parameters of the instances and the path are sent to a PHP script. This script saves the parameters of the new path in the database and sends a confirmation back to the user.

The gates interface allows the user to create, modify, and delete connections to foreign scenarios. In this case, a PHP script generates a VRML file including the geometries of all existing scenarios with their paths, and the tools needed to switch between the different spaces, visit the scenarios and create the gates. All of the spaces and scenarios were put into one file, because this enables one to jump between different scenarios without having to reload each file from the server. Once the gates are defined, they are saved in the same database as the scenario. In a second step, a library with the objects of the other authors can be visited and an object to decorate each gate can be selected. This object can be an image, a video clip, or a three-dimensional model.

The explore interface allows for the exploration of the scenarios and the hyperstructure. As in the previously described interfaces, a PHP script generates a VRML file that contains all the objects, the instances and the path, and enables navigation through the scenario. In this interface, the navigation mode of the VRML world is set to 'none' so that the only way to navigate inside the space is by following the path created by the scenario's author.

3. THE [ROOMZ]&[CONNECTIONZ] COURSES

[roomz]&[connectionz] have provided the online environment for two courses about architecture and new media. The two courses spanned a period of six weeks each and were taught to 130 second-year architecture students. The students were able to work from home as well as on the university campus.

A digital model of Georges Vantongerloo's sculpture, "Rapport des Volumes" (1921), provided the spaces for the scenarios. This sculpture is composed of eleven L-shaped volumes with different proportions. The students had to select three adjacent volumes as the geometry for their scenario. Working with abstract spaces that lack scale and orientation led to a freer interpretation of the spaces and the definition of their architectonic qualities by the scenarios themselves. (Strehlke, 2001)

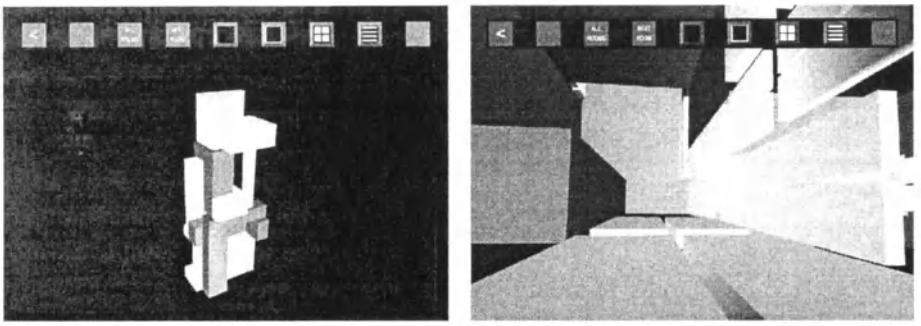


Figure 3. Interface for selecting volumes from Georges Vantongerloo's sculpture. Inside the volumes, different types of objects can be selected.

An additional three-dimensional interface was provided to allow the student to select the volumes in a playful, intuitive way and to enter the space within them. Wherever two volumes touched, the type of opening between them had to be defined, either as a full opening, a frame, four blocks, or with slats. The interface supported the search for the openings, because only after each one was defined, was it possible to save the composition of spaces. The sculpture from Georges Vantongerloo offered a large variety of possible compositions with different characteristics. Some students composed a circular arrangement, while others aimed at a linear sequence; some compositions had an introverted character while others had many openings to the outside. The colors defined in the myscenario interface further enhanced the individual character of each composition.

3.1 Spacepixels, Re-Space, Actors in Space, and Motion&Motion

Four specific themes about architecture and new media - light, transformation, dynamics, and motion - were chosen as possible fields of concentration for the students. The respective teaching modules were called Spacepixels, Re-Space, Actors in Space, and Motion&Motion. The students had to choose one module each semester. They were allowed to select the same one twice in order to deepen their understanding of the theme and the related technologies.

Spacepixels explored the dialog between light and material as an architectural phenomenon. The influence of daylight and artificial light in a virtual model was analyzed and images with different light qualities were produced. The two-dimensional images were placed in the space or on the walls to create enhanced or ambiguous readings of the space.

Re-Space dealt with the representation, perception, and transformation of space. This module worked with the duplication of space and aspects between the original, the transformed, and the animated model of the space.

Actors in Space involved placing dynamic objects into the space. These objects could be seen as digital organisms, animated geometry, and visual statements that established relationships with the space by introducing aspects of orientation, scale, and meaning.

Motion&Motion focused on the combination of two kinds of motion: Motion shown by animations and motion through the space. Animations and videos were retrieved from the Internet, created directly from the digital model of the space itself, or recorded with a video camera and then placed in the space.

In addition to the chosen module, the students were confronted with the themes and the design possibilities of the other modules by being able to view work-in-progress through the common online environment and reviews in class.

3.2 Scenarios and Hyperstructure

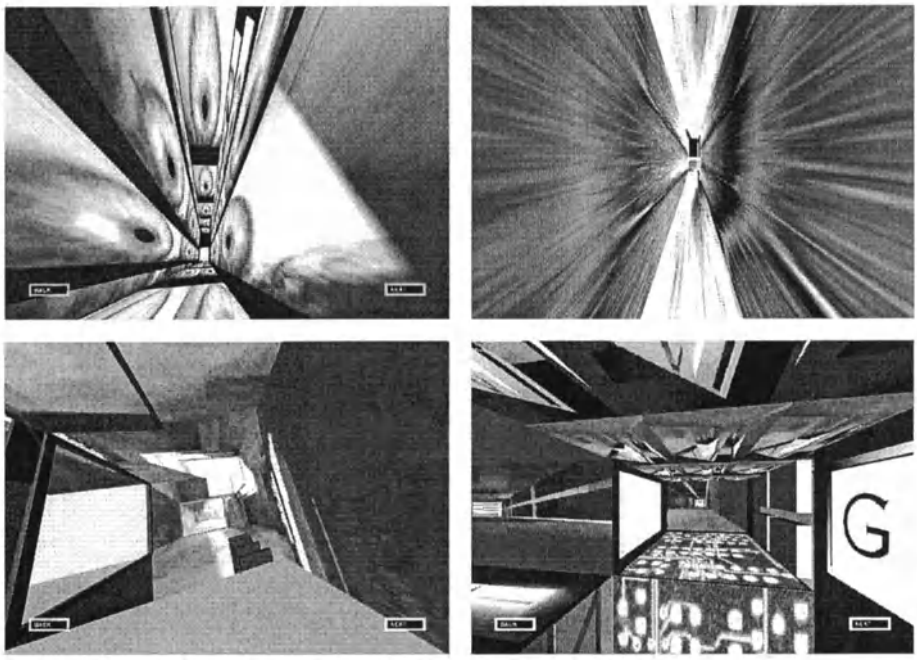


Figure 4. Four examples of student works: Eyes tracking the visitor, images enhancing the motion, renderings extending the virtual space of the sculpture and images serving as gates to other scenarios.

In the first course, [roomz], the students were asked to create a scenario. They started by selecting the spaces from the sculpture and a module and then created the visual objects they wanted to place inside their spaces. After placing the objects in the spaces, they designed a path through the geometry.

Although the students tended to very carefully create the visual objects and place them into the sculpture, the time provided to create the path proved to be a little short, leading to very simple paths mostly composed of viewpoint sequences that visited all the instances placed inside the spaces. Only a few groups worked in a conceptional way, exploring the possibilities of designing the path itself. They designed a motion inside the space by defining the time between the scenes, changing the viewing angle between the scenes, and defining the orientation for each scene. The end result of the first course was a collection of visually appealing scenarios inside the same sculpture but without any connections between them.

The goal of the second course, [connectionz], was to create a hyperstructure that contained the scenarios from the first course as well as new ones. The students were again asked to compose their own space from Georges Vantongerloo's sculpture, choose a module, and create an architectural experience in the selected space. To put more emphasis on the motion through the space, the path had to be created before placing objects inside the spaces. As a result, interesting ideas were developed for the motion through the spaces and the path gained relevance in the overall design of the scenarios.

In addition, each new scenario in [connectionz] had to be connected with one or more other ones with gates.

3.3 Gates into Completed versus Changing Scenarios

A gate is a link between two scenarios. When the students created a gate, they had to select the portion of the path in the linked scenario that the gate should connect with. A specially developed interface allowed them to easily select the entry and exit points of the path in the foreign scenario. In addition, an image from the foreign scenario had to be chosen to represent the gate.

The different strategies invented by the students to connect the scenarios proved to be very interesting: They involved the relations between different geometries, motion, and the objects of the connected scenarios.

The students could create gates to completed scenarios from the previous course as well as to scenarios being created in the current course. If a scenario from the current course was chosen, the connection was not stable, since the path as well as the appearance of the linked scenario could still be altered. The ability to connect to a scenario-in-progress gave the students the

opportunity to work in an interconnected, dynamically changing environment - an experience only a few were eager to make. Many students connected their new scenario to their scenario of the previous semester.

In the case of a connection to a scenario-in-progress, the authors sometimes decided to communicate over email to coordinate their efforts and keep the other informed about possible future alterations.

Another special issue leading to interesting strategies was the decoration on the gate, which had to be chosen from the library of visual objects of the linked scenario. Some students did not want to place any of the foreign objects in their scenario. So, in order to control the appearance of the gate, they emailed an object to the other group and asked them to upload the image into the database so that they could select this image as the representation of their gate.

4. EVALUATION AND CONCLUSIONS

The goal of this work was to create an environment to express architectonic ideas in and of digital spaces by offering intuitive and widely accessible interfaces. In the following discussion, different aspects will be considered: The focus and limitations of the tools, composing in the myscenario interface, and the results of the described course in architecture and new media.

4.1 Focus and Limitations of the Tools

The tools and interfaces can be used to create three-dimensional scenarios and visual objects can be imported, but the geometrical shape of the given spaces cannot be influenced in the current version of the interfaces. The scenarios themselves primarily allow navigation through the given hyperstructure but only very limited interactivity. For the designer as well as the viewer, these limited possibilities allow one to very quickly master the interface and use it in an intuitive way. Additions to the current functionality are possible within the current interfaces or by adding new interfaces, but require a careful rethinking of the interface concept.

The results from the courses show that the current interfaces already offer great potential. While some of their limitations enhance ease of use and creative work in the three-dimensional realm, there are others, inflicted by the software, the hardware or the interface paradigm, which hinder creative efforts. One of the major disadvantages is the need to click on movies in order to start them. Ideally they should start on their own, but even our most powerful computers struggled with this task. In addition, moving through the

space can only be accomplished by hitting the 'next'-button to reach the next viewpoint. Ideally it should be possible to design a smooth motion without stopping at every viewpoint, rather using stops at points where the user can make a decision or interact in other ways. This is again a problem of computing power. In the myscenario interface, a method of manipulating instances that is more direct than using control panels would be desirable. However this is currently not possible, because the interaction paradigm is dictated by VRML and is, in this regard, hard to overcome.

4.2 Composing in the myscenario Interface

The myscenario interface was created to allow the designer to create a scenario that presents interesting aspects of the space to the reader. In the course, we discovered that it has even far greater potential. First of all, it allows the students to explore the space at hand and interactively explore its qualities. It challenges spatial thinking, something that happens mainly in the process of placing visual objects and defining the appropriate viewpoints. This turned out to be a non-trivial task, since the motion to and from the viewpoint had to be taken into account as well. The introduction of the objects also led to the discovery of new qualities of the space. Finally, the design of the whole scenario demanded for a story-like structure and an idea of the future audience; the result can be considered a narrative in space. The step of creating gates into other scenarios is similar to making cuts in movies, but is less controllable.

4.3 Results from the Architecture and New Media Course

The 130 students that participated in the two courses helped us discover many of the flaws and qualities of the interfaces. The flaws were immediately improved during the courses and the qualities pointed out during the lectures and individual sessions with the students. This led to interfaces with a high level of consistency and usability even though they were developed in a very short period of time. In the end, they profited tremendously from the collective exploration.

The teaching concept of the [roomz]&[connectionz] courses led to a variety of learning experiences. Architecture, multimedia, and the connected environment were the central theoretical issues. Learning how to use software applications was a pragmatic necessity in order to be able to adequately express ideas. The three-dimensional interface was an important development for this course, taking the architectural discourse to a higher degree, from digital images to digital scenarios, and creating new challenges

for the students. The resulting hyperstructure of the [roomz]&[connectionz] course sequence shows the great potential of this approach.

The results also showed very appropriate use of the media. One of the greatest challenges was to make sound choices out of the endless possibilities. Often the final scenario included only a small selection of the material that was produced in favor of a clearer but nevertheless interesting story. Also the hyperstructure, the common achievement of the whole class, gained qualities thanks to decisions which led to the design of clearer and more unique scenarios. Vantogerloo's sculpture, which in the beginning of the course consisted of 11 empty volumes, became a powerful, multithreaded narrative space containing the thoughts of 130 authors.

The [roomz]&[connectionz] hyperstructure can be explored at http://alterego.arch.ethz.ch/connectionz.

5. ACKNOWLEDGMENTS

We would first like to thank all of the students. They created fantastic work. Special thanks also to the team for their contribution to the course: Fernando Burgos, Patrick Sibenaler, Maria Papanikolaou, Fabio Gramazio and Cristina Besomi.

6. REFERENCES

Anders, P., 1998, *envisioning cyberspace*, McGraw-Hill, New York, 1998, p. 86.

Colafranceschi, D., 1995, *Architettura in superficie : materiali, figure e tecnologie delle nuove facciate urbane*, Gangemi, Roma,

Engeli, M, 2000, *Digital Stories ThePoetics of Communication*, Birkhäuser, Basel, p. 63.

Kurmann D., 1998, "Sculptor - How to Design Space", *Proceedings of* CAADRIA '98, Osaka, p. 317-325.

Mitchell, W. J., *City of Bits,* MIT Press, Cambridge.

Murray, J. H., 1997, *Hamlet on the Holodeck – The Future of Narrative in Cyberspace*, The Free Press, New York.

Rushkoff, D., 1996, *Playing the Future – What we Can Learn from Digital Kids*, Riverhead Books, New York.

Schank, R., 1995, *Tell me a Story – Narrative and Intelligence*, Northwestern University Press, Evanston, Illinois.

Schmitt, G., 1999, *Information Architecture*, Birkhäuser, Basel Bosten Berlin.

Strehlke K, 1999, "An environment for collaborative three-dimensional modeling over the internt", *Proceedings of Avocaad second international conference*, Hogeschool Voor Wetenschap enKunst, Brussels, p. 63-68.

Strehlke K, 2001, "[roomz] and [connectionz]", in: M. Engeli (ed.) *bits and spaces*, Birkhäuser, Basel Bosten Berlin, p. 88-99.

Tschumi, B., 1995, *Event-cities*, MIT Press, Cambridge.

The Role of Place in Designing a Learner Centred Virtual Learning Environment

Steve Clark and Mary Lou Maher
University of Sydney

Key words: virtual place, virtual learning environments, design studio

Abstract: There are numerous approaches and tools for creating a virtual learning environment. The most common approach is to provide a repository of learning materials on a network to facilitate the distribution of the course content and to supplement this material with communications software such as email and bulletin boards. In this paper we highlight the role of place in creating a learning experience and its relevance to current learning theories. Architectural design becomes an important consideration when virtual learning environments are considered places. We present a framework for guiding the development of virtual learning environments that shows how place and design can combine with learning theories and technology. Finally, we draw on our experience in developing a virtual campus and virtual design studios to demonstrate how place and design can be the basis for virtual learning environments.

1. INTRODUCTION

Virtual Learning Environments are built on a foundation of two key elements: computer technology and education. The technology aspect of virtual learning environments provides facilities for learning management tools, on-line learning frameworks, collaborative learning environments, web course design tools, etc. The software typically resides on a server and is designed to manage or administer various aspects of learning; delivery of materials; student tracking; assessment; and so on (Milligan, 1999). Though predominantly found using on-line technology virtual learning environments are not restricted to this domain, but utilise computing technologies afforded

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 187–200.
© 2001 *Kluwer Academic Publishers.*

by CD-ROM and more recently DVD-ROM. Many virtual learning environments also combine the two technologies, on-line and CD-ROM.

These two technological environments, not withstanding current technical capabilities such as storage and data transfer, have two distinct differences. On-line environments are dynamic and operate in a state of flux, subject to change at any given time. This ability to document change quickly makes them an ideal environment for communication and management. In juxtaposition, CD-ROM and DVD-ROM environments are fixed, and stable in terms of change. They provide an environment that can store fixed resources and content, which like books have to be edited and reproduced to document the new changes and conditions contained within their environment.

Traditionally, virtual learning environments were simplistic and crude, containing lots of text and simple graphics, with very little dynamic interaction. The design and development was influenced and carried out by programmers and subject matter experts. These were little more than electronic books containing the lecturer's notes. As computing technology advanced the capability to provide more sophisticated and dynamic applications for both CD-ROM and on-line environments increased. The design and development process saw the introduction of graphic designers who made the environments visually more engaging. Learning technologists helped to balance the influence of the programmers on the learning environment and bring the focus back to the pedagogical issues.

Today, we have the ability to create very sophisticated and complex interactive virtual environments. Virtual environments are not restricted to just text and graphics, but can include sound, video, and animation, all possible on both CD-ROM and on-line. These virtual environments are populated with communities, which are able to interact and communicate with each other in many forms. These virtual environments have the shapes, form, structures and functionality that are akin to the physical world. It is appropriate then to consider the role that architectural design can take in the design and development of virtual environments.

In this paper, we consider the importance of place in learning and present a framework for guiding the development of a learner centred virtual environment. We justify the use of place in learning with a constructivist view of learning and bring the focus back to the learner as the centre of the virtual environment to provide them with a learning experience. By developing the sense of place as the core of a virtual learning environment students are able to construct external representations of their knowledge in a manner similar to creating artifacts in a physical room. In this paper we present how the role of place can influence the design of virtual learning environments.

2. A FRAMEWORK FOR VIRTUAL LEARNING ENVIRONMENTS

Virtual Learning Environments are inevitably designed with a pedagogical model in mind, that is not made explicit (Britain and Liber, 1999).

The development of virtual learning environments is typically guided by the consideration of two key elements: Technology and Education as shown in Figure 1.

1. **Technology**: is made up of many sub-categories based on computing technology.
2. **Education**: is made up of many sub-categories based on educational models.

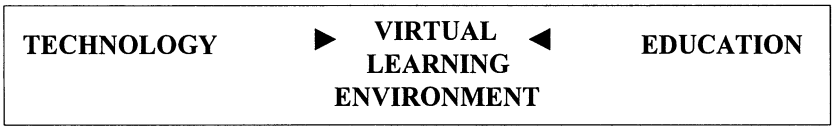


Figure 1. Traditional framework for virtual learning environments

In previous research by Clark (Clark et al, 1998, Clark et al, 1999), it was found that using this traditional framework can aid in the designing of virtual learning environments. Murder under the Microscope - Professional Challenge 98 was a hybrid multimedia CD-ROM and on-line web-based virtual learning environment (see Figure 2). The learning environment was based on a 'REAL', rich environments for active learners. REAL's are comprehensive instructional systems that engage students in dynamic, authentic learning activities that increase their control and responsibility over the learning process while they learn problem solving and collaborative skills and content (Dunlap and Grabinger, 92. Grabinger, Dunlap and Duffield, 97).

Professional Challenge 98 was targeted at a broad section of the educational community from schools to universities, communities and businesses, and aimed to create an awareness of different environments (town, forest, crops, industrial, coast) and human impact on them. Teams of students worked to determine the impact of a diverse range of problems – dam proposals, farming, urban development and tourism – and propose alternative solutions. Each team was able to specify the problem(s) they wanted to tackle(see Figure 3), their level of involvement (simple to complex) and the time they could spend on the tasks (Clark, 98, Grabinger, 98).

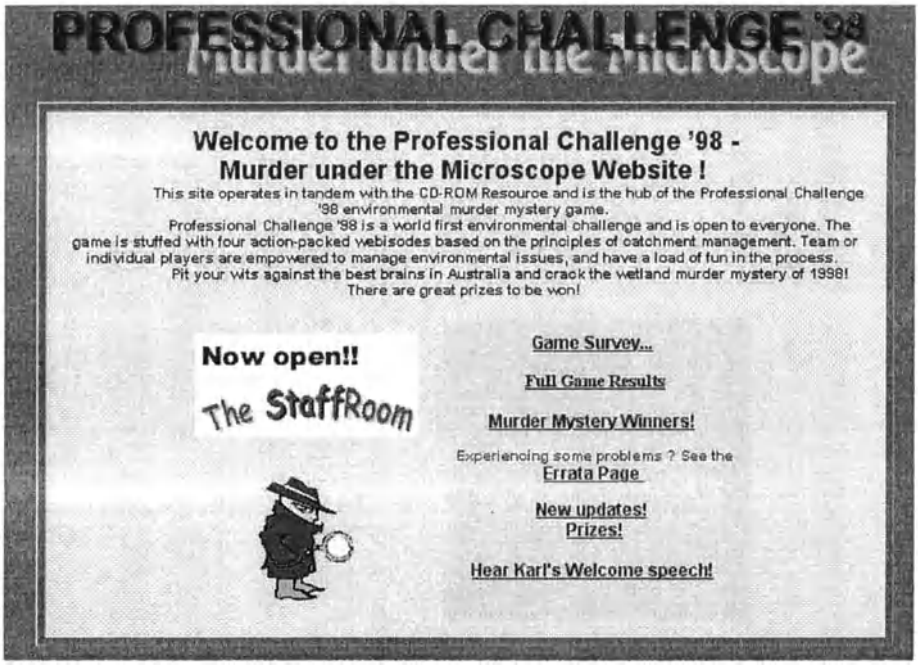


Figure 2. Murder under the Microscope Professional Challenge 98.

Professional Challenge 98 took over six months to design, develop and implement. In the beginning, the multidisciplinary project group worked in isolation from each and the project was driven by technological capability. The introduction of the framework with both technology and educational concerns brought the project back into alignment and provided a clearer sense of perspective. At first the project was driven by technology with pedagogical issues secondary. The framework brought a shift in the group focus and balanced the focus between technology and education. With a clearer pedagogical direction and how technology could best deliver, the group as a whole became more cohesive.

Professional Challenge 98 provided an excellent example of a "REAL", promoting study and investigation within authentic contexts, encouraging student responsibility, initiative and decision-making while cultivating a collaborative learning community. Unfortunately, what Professional Challenge 98 lacked was a sense of place. Participants in the challenge were aware that other competitors were competing and could chat to experts and other teams on-line, but these events were still in isolation and alienating with no metaphorical references to physical embodiment. The challenge and its environment lends itself ideally to the possibilities that are afforded by creating a sense of place.

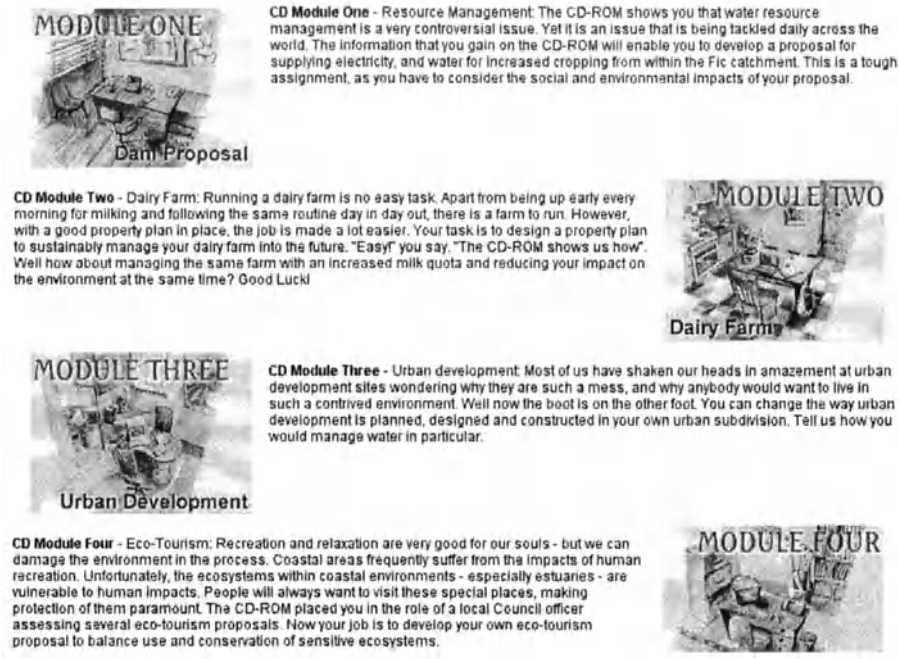


Figure 3. Projects web-page for Murder under the Microscope.

Focusing on supporting the learner and the learning environment the traditional framework does not make explicit the importance of the learning experience and the context in which the learning occurs.

We propose a framework for virtual learning environments that considers the learning experience and draws on design as a pedagogy, illustrated in Figure 4.

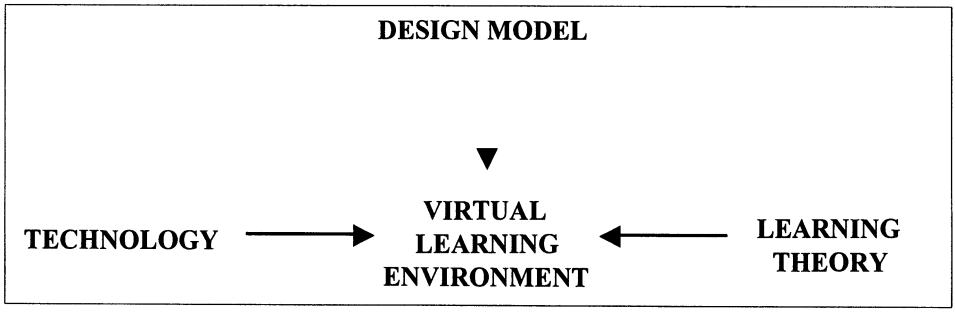


Figure 4. A framework for virtual learning environments that considers the importance of design

The development of this framework leads to a model for virtual learning that can benefit from architectural theories and principles of design teaching.

Figure 4 illustrates the framework as having a third component, design models. Design models bring relevant design theories and pedagogy to the learning environment, focussing on the role of context and experience.

Where each of the components of the framework in Figure 4 can be elaborated to take into account numerous alternatives, we focus on a specific style of virtual learning environments. In Figure 5 we show how the technology aspect of a learning environment can be supported by a virtual world, the learning theory that we apply is constructivist, and the design model that we consider important for learning is situatedness.

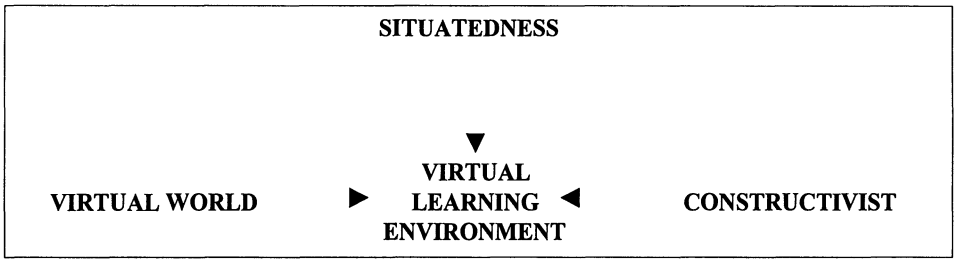


Figure 5. Model for designing a learner-centered virtual learning environment

This style of virtual learning environment supports the four key processes fundamental to a constructivist learning environment identified by Jonassen and Duffy: context (meaningful and authetic), construction (of knowledge), collaboration and conversation (between student/student and teacher/facilitator/mentor) (Jonassen et al, 1995. Duffy and Cunningham, 1996. Jonassen, 1994). The role of place in a learning environment is to provide the basis for these key processes. The place is designed specifically for the context. For example, in teaching a web site design course the virtual place is a gallery of student work. The place is also designed explicitly to support the construction of knowledge, where the student has the ability and is expected to contribute to the place. For example, in teaching web site design the students add to the gallery with their own exhibitions. And the use of virtual worlds intrinsically support collaboration and conversation.

Situatedness is the notion that addresses the role of the context. This concept is founded on the work of Bartlett (1932) and Dewey (1896) and has recently been introduced into design research by Gero (1998). Situated design considers the context in which the designing occurs as an important component to the design process and the knowledge available to the designer. When using this approach to learning and designing, the virtual learning environment provides the tangible situation in which the learning occurs. The design of the environment as a place takes this into consideration, resulting in a specially designed place for each learning experience.

Situatedness and constructivism are mutually compatible by their historical connection to cognitive psychology and it is this connection that makes them an appropriate choice in designing a learner-centred virtual learning environment.

Virtual worlds, such as Active Worlds (http://www.activeworlds.com) and VWorlds (http://www.vworlds.org), are networked environments that create a sense of place and a sense of presence of others in the place. They are inherently collaborative especially when populated by other people. Selecting a virtual world as the technology for a virtual learning environment provides the basis for a place, but the place itself needs to be designed. The design of the place can be influenced by architectural design.

3. CONSTRUCTIVISM AND THE ROLE OF PLACE

By understanding the fundamental nature of constructivism, we see how developing a sense of place can enable learners to construct their own learning. Constructivism asserts that we learn through a continual process of building, interpreting and modifying our own representations of reality based upon our experiences with reality (Jonassen, 1994). Three distinct characteristics of constructivism are:

1. Knowledge is not a product to be accumulated, but an active and evolving process in which the learner attempts to make sense out of the world (Gurney, 1989).
2. People conditionalise their knowledge in personal ways (Gurney, 1989). That is knowledge is referenced to a context in which it was encountered, and so can be applied spontaneously in new situations.
3. Learning happens within a social and collaborative context. Conceptual growth comes from sharing perspectives and testing ideas with others, and modifying internal representations in response to the process of negotiation with peers and teachers (Bedner et al, 1991).

All three of these characteristics emphasise the need for an environment, which will allow context, collaboration and communication. The Virtual World is an ideal three dimensional environment to develop a constructivist virtual learning environment where students are provided with a sense of place and context, and are able to explore, build and share their learning experience.

4. IMPLEMENTING PLACE IN A VIRTUAL LEARNING ENVIRONMENT

The development of place for learning environments has resulted in classrooms, laboratories, and studios. The design studio, a critical aspect of education in an architecture curriculum, assumes a place for students and design teachers to come together in a learning context. We take this approach as a basis for establishing place in a virtual learning environment and extend it to take advantage of the possibilities that software technology offers that is not possible in a physical studio.

In developing virtual design studios, the desktop metaphor and the place metaphor are compared in Maher and Simoff (1999). The desktop metaphor refers to the use of collaborative tools as if they were lying on a working desk of a physical office. On the desktop, and nearby, a designer finds tools for drawing (eg. pencils, rulers), communicating (eg. telephone), archiving (eg. folders, filing cabinets), organising (eg. diary), finding information (eg. catalogues, archives), and so on. In general, he has access to all the office resources apt to perform the design task. On the electronic desktop – which is based on a metaphor of the physical one - all the functions are collected on the same interface, in this case, visible on the computer screen. This approach is the most common and is presented as the "toolkit approach" in Lin and Protzen (1997).

Virtual places, which include virtual worlds and virtual reality applications, are *computer-mediated dynamic world models that create a sense of place*. The Internet has been accommodating more than a dozen different technologies supporting multi-user text-based, and two- and three-dimensional graphical virtual worlds. When adopting the place metaphor, preparing a virtual design studio is much like designing a physical studio (Maher et al 2000). The studio is set up to facilitate and support collaborative design activities. A virtual design studio differs from the physical design studio in a significant way: where a virtual studio can automatically react to the people's use and presence, a physical studio is passive and is changed only when people physically change it.

Figure 6 shows a virtual design studio implemented in Active Worlds. The students participate in the studio by going to the virtual place to present, discuss, and develop their designs. In addition to the internet facilities of web pages, bulletin boards, and email, the students have the experience of walking through the studio as avatars and meeting other students. The virtual design studio as a virtual world becomes a place for having a learning experience, in contrast to the use of web pages for managing a learning experience.

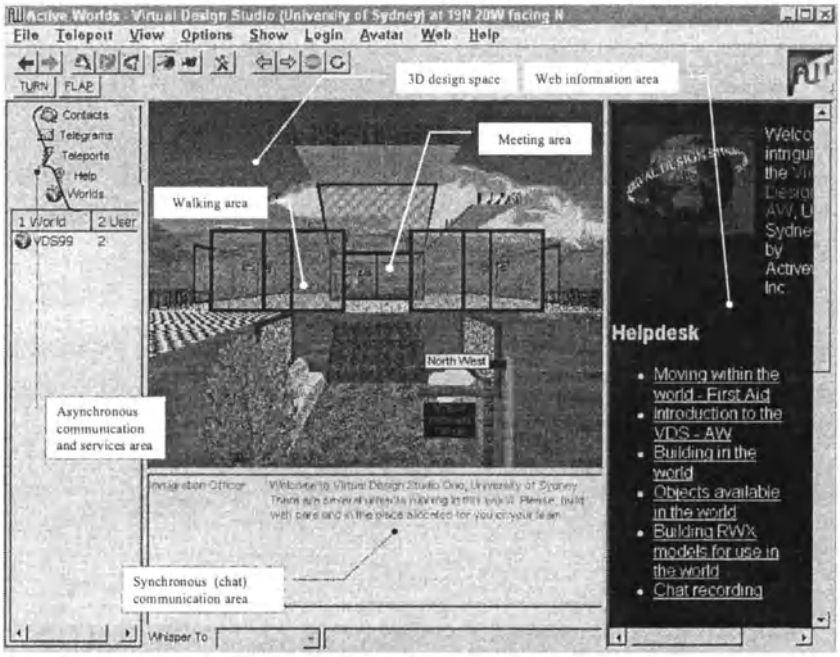


Figure 6. A virtual design studio as a learning place.

More generally, we have developed a virtual campus approach to virtual learning environments, where the emphasis is on collaboration and the learning experience (Maher, 1999), shown in Figure 7. There are 4 parts to the interface:

1. The representation of the place as words or 3D models.
2. The icons for access to information or other learning environments specific to a learning task.
3. The text of the conversation and activities in the room.
4. The typing area for commands or discussion.

The current implementation structure of the Virtual Campus is shown in Figure 8 (Maher, Clark and Simoff, 2001). It is based on two separate servers: place server and course server. The place server is based on a lambdaMOO server with the BioGate interface between the MOO database and the web server. In the MOO server every participant is represented by a character. The course server, where the course materials reside, is based on WebCT courseware server. The bridging data interface passes the information about the character and current location in the place to the course server. In the user interface this is reflected as an additional icon on the toolbar. Thus, to access the course materials corresponding to a room in the learning space of the Virtual Campus, a student selects a "book" icon in the toolbar.

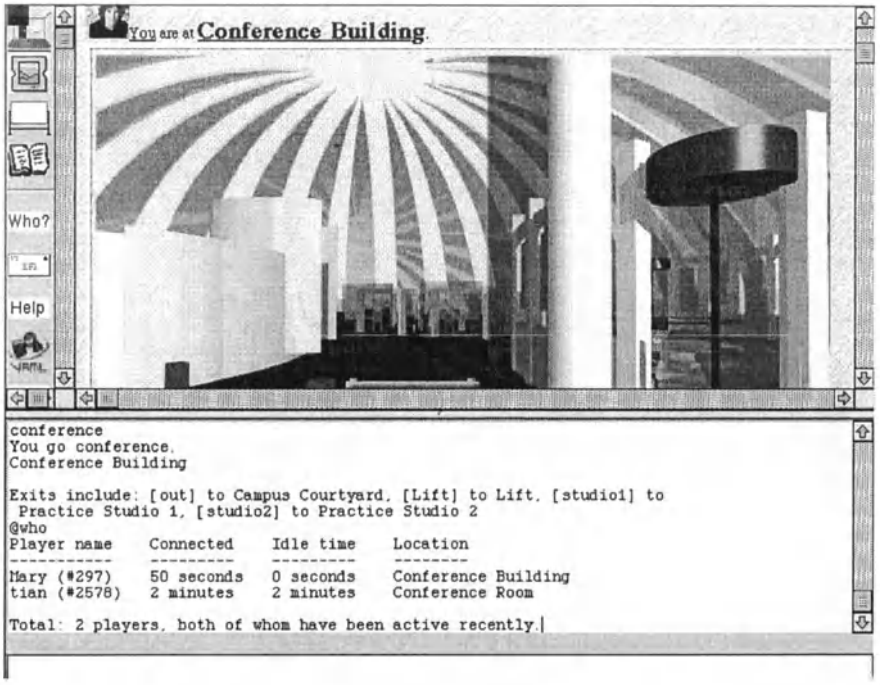


Figure 7. The Virtual Campus as a place

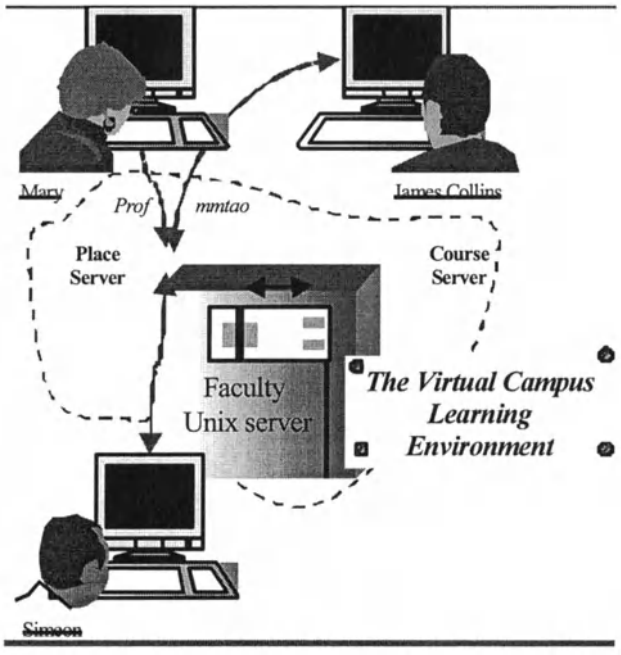


Figure 8. The implementation structure of the virtual campus

The loose integration approach offers an open learning space. The idea of the open learning environment is the incremental addition of technologies to the learning environment in a way that doesn't violate the consistency of existing virtual space organisation and human-computer interaction interface. In terms of implementation, this means the development of a bridging interface and the design of the corresponding icon for the toolbar. For example, we use ActiveWorlds as a virtual place for certain learning activities. Under the "loose integration" framework, the integration with ActiveWorlds requires a similar bridging data interface which passes the information about the character and current location on the place server to the ActiveWorlds server.

We illustrate the role of place further by describing a course on web site design. Previously this course has been taught in a virtual learning environment that provided course materials and an online bulletin board. The course materials were available to each student on web pages with password protection and access to the materials was monitored and available to the instructor.

This year, in addition to the course materials, we prepared a gallery in a virtual world as the place in which the students prepared and presented their assignments. The gallery was designed in Active Worlds and is shown in Figure 9. The gallery area is used as a meeting place for the students and lecturer of the course and in this open forum reminiscent of the acropolis, where philosophical debate takes place. The students and lecturer are able to convene in a common meeting place and share ideas, discuss problems and answers, and work collaboratively giving peer and lecturer support on each others' designs. Each student has their own personal name plate acting as a hyper link to their design web pages. These links enable anyone at any time to review the progress taking place in the students design portfolio. This ability to review and discuss together is an important feature in the process of "learning by designing". The other name plates are hyper links to the course, content and resources. The lecturer's name plate is linked his web page and the course information and notice boards. Another link takes students to the Web-CT site where learning activities and asynchronous discussion boards are available.

The virtual gallery provides a sense of embodiment and students choose an avatar to represent themselves in a virtual learning environment. The response to this type of learning has been very positive with students able to communicate more freely and openly, where they would otherwise have been more reserved and less inclined to give expressive opinion on other students work.

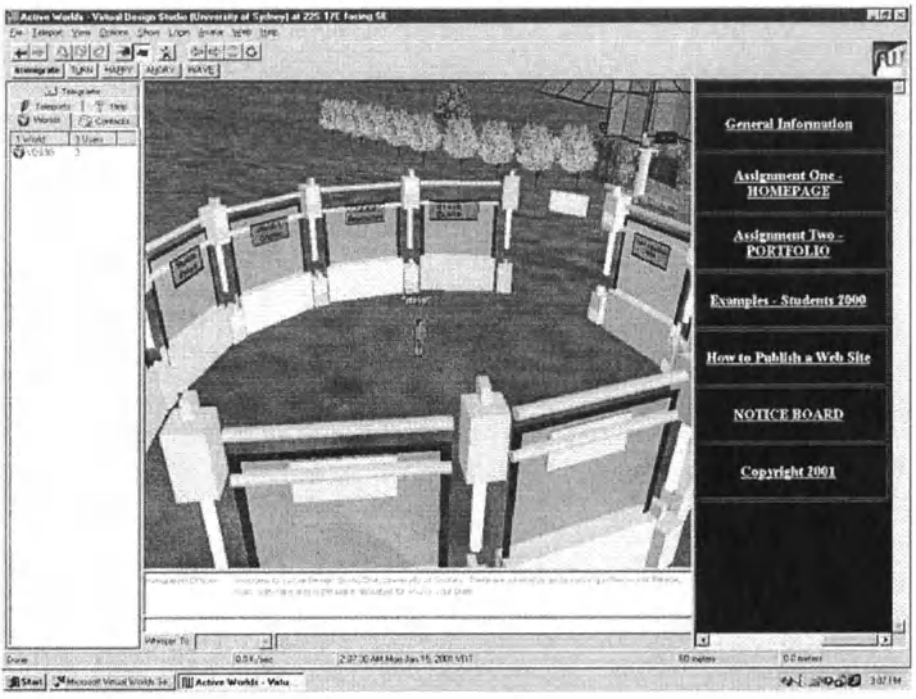


Figure 9. Gallery for meetings and presentation of assignments

5. SUMMARY

Incorporating place in virtual learning environments has pedagogical significance as well as encourages community and collaboration. The development of virtual design studios and virtual campus environments have not emphasised the role of place nor has there been a significant contribution from architectural design. In this paper we highlight the importance of the context or situation in which learning occurs and how a sense of place in a virtual learning environment can provide this context. We have developed place environments for various learning experiences and present a framework in which the design of virtual learning environments can explicitly address the role of place and context

6. REFERENCES

Bartlett, F. C. (1932 reprinted in 1977) Remembering: A Study in Experimental and Social Psychology, Cambridge University Press, Cambridge.

Britain, S. and Liber, O. (1999) 'A Framework for Pedagogical Evaluation of Virtual Environments' JTAP Report 041. http://www.jtap.ac.uk/reports/htm/jtap-041.html

Bednar, A. K., Cunnignham, D., Duffy, T. M. and Perry, J. D. (1991), 'Theory into practice: how do we link?', in Anglin, G. J. (ed.), Instructional Technology: Past, Present, and Future (pp. 88-101), Englewood CO, Libraries Unlimited.

Clark, S., Koppi, A.J., Chaloupka, M.J. and Westbrook, J. (1998) Building the Virtual Classroom: Utilising the Internet to provide a backbone for external assessment and evaluation. Association for Learning Technology Conference, University of Oxford, Oxford, UK, September 21-23.

Clark, S., Koppi, A.J., Fenton-Kerr, T. (1999) Designing Learner Centered Environments. Association for Learning Technology Conference, University of Bristol, Bristol, UK, September 21-23.

Clark, S. (1998) 'Discussion workshop - REALs and Professional Challenge 98' ALT-C 98, University of Oxford, UK, September 21-23, 1998.

Dewey, J. (1896 reprinted in 1981) The reflex arc concept in psychology, Psychological Review 3: 357-370.

Duffy, T.M., and Cunningham, D.J., (1996). Constructivism: Implications for the design and delivery of instruction. In D.H. Jonassen, (Ed) Handbook of research for educational communications and technology, N.Y; MacmillanLibrary reference USA.

Dunlap, J. C., & Grabinger, R. S. (1992). Designing computer-supported intentional learning environments. Paper presented at the Annual Conference of the Association for the Development of Computer-Based Instructional Systems, Norfolk, VA.

Gero, J. S. (1998) Conceptual designing as a sequence of situated acts, in I. Smith (ed.), Artificial Intelligence in Structural Engineering, Springer, Berlin, pp. 165-177.

Grabinger, S., Dunlap, J.C. and Duffield, J.A. (1997), Rich environments for active learning in action: *problem based learning.* ALT-J, Vol 5, No 2.

Grabinger, S. (1998) REAL's for ALT-C, *Active Learning, Technologies for Lifelong Learning*, page 59, No. 9. December 1998.

Gurney, B. (1989), *Constructivism and Professional Development: A Stereoscopic View*, ERIC Document ED 305 259, 1-28.

Jonassen, D.H. (1994), Thinking Technology: Toward a constructivist design model. Educational Technology, 34(3), 34-37.

Jonassen, D. H. Davidson, M., Collins, C., Campbell, J., and Haag, B.B., (1995). Constructivism and Computer-Mediated Communications in Distance Education. The American Journal of Distance Education, 9(2), 7-26

Lin, T-H. and Protzen, J-P. 1997. Desktop design: a toolkit approach to collaborative design, in M.L. Maher, J. Gero, and F. Sudweeks (editors) *Formal Aspects of Collaborative Design*, University of Sydney, pp 237-251.

Maher, M.L. (1999). Designing the virtual campus as a virtual world, Computer Supported Collaborative Learning (CSCL99), pp 376-382.

Maher, M.L., Simoff, S., and Cicognani, A. (2000). Understanding Virtual Design Studios, Springer-Verlag, London.

Maher, M. L. and Simoff, S. (1999) Variations on a Virtual Design Studio in J-P. Barthes, Z. Lin and M. Ramos (eds) Proceedings of Fourth International Workshop on CSCW in Design, Universite de Technologie de Compiegne, pp.159-165.

Maher, M.L., Clark, S. and Simoff, S. (2001) Learner Centred Virtual Environments as Places, in Proceedings of the European Conference on Computer Supported Collaborative Learning, to appear.

Milligan, C. (1999) The role of VLEs in on-line delivery of staff development. JTAP Report 573. http://www.icbl.hw.ac.uk/jtap-573

WebCT: http://www.webct.com/

Professional Challenge 98 web-site: http://www.itl.usyd.edu.au/pro/

Virtual Campus: http://www.arch.usyd.edu.au:7888

Games in Early Design Education

Playing with Metaphor

Robert F. Woodbury, Susan J. Shannon and Antony D. Radford
Adelaide University

Key words: Design, learning, play, games, metaphor, computer-aided design, digital media.

Abstract: Play and design can be put into metaphorical relation. To do so is to let each inform the other. As part of a larger project, we have used the metaphor of play in creating and using learning resources for early design education. In doing so, it became apparent that the entailments of play, the other metaphors that both frame and are framed by play, needs to be better understood. We discuss seven metaphors related to play: games, exploration, balance of forces, tactility, intrinsic reward, embodiment and rules and how we use these in learning games.

1. INTRODUCTION

> *A society concerned with producing a fair number of creative and imaginative adults must protect the play modalities of thinking developed in childhood."* (Bower, 1974, p8)

As part of a nationally funded project, we have developed and used "games" as student-centred teaching resources to enrich the capacity for design in beginning students in architecture, landscape architecture and urban design. Students are encouraged to learn inter-actively in a milieu characterised by self-directed play in a low-risk computer modelling environment. This paper explores the cognate ideas around play—the aspects of play that make it an effective metaphor for early design education.

Traditionally, in design studios, students have designed manually at drawing boards, utilising their intrinsic skills of inventing, manipulating and describing *form* (which is used throughout this paper as the expression of

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 201–214.
© 2001 *Kluwer Academic Publishers.*

designed spatial structure) to move towards a design solution that satisfies the requirements of the project. For those students who lack confidence and skill in form-making, their inability to invent, manipulate and describe form is a debilitating handicap leading to frustration and disillusionment. In a traditional design studio, the passage of time is a major factor in gathering skills at inventing and manipulating form, whilst the skills to describe form graphically and orally gradually accumulate. For many students this process is fraught: some leave, some under perform for years.

In studios in which digital media dominate, these givens can be overturned, due substantially to the ability of the mechanism of CAD modelling to interpose into the process of making, inventing and describing form. Our contention is that bringing CAD-supported form-making operators to the fore in formal lectures and reiterating these lessons with practice in an enjoyable, low-risk social learning environment can build both the confidence to continue and the competence to perform.

2. WHY GAMES? WHY PLAY?

Games have good press. They are fun. They are social. The outcomes do not really matter in comparison with the joy of playing them —"It's just a game!"—"We're not playing for sheep stations here!" They are culturally universal. They are ageless. They are frequently rule-related—they pit the player against the rules to produce a good outcome. When rule free, they are open-ended, and encourage inventiveness either prior to or during play.

In developing the games that form the basis for the teaching and assessment tasks in this subject, we initially thought that "play" and "games" captured and developed the essence of "good" designing. Experienced designers engage play and reflection as principal tools in their designing repertoire, whilst frequently relying on informal rules for form-making, which results in recognisable styles of individual and collaborative work. Our hope was that we could bring these aspects of mature designers into the early learning fray. A pilot evaluation using a participant-observer methodology gave us confidence to continue. A comment from a student reads:

> *It's the most enjoyable, relaxed (subject), but the work is fun to do, unlike some other subjects...*

This comment supports the assertion that the games are fun to play. Another commented on the use of games in the tutorials that:

> *Games in Tutorials helped really understand what we were doing (not only how).*

This comment attests to our efforts to create games that are both intrinsically interesting and relevant to learning about design.

After several years of working with learning tasks we describe as games, we find that we are playing with wider aspects of play than are commonly received. The existing literature on the topic, for example, the large games in education literature that focuses almost exclusively on role-playing games, and, in architecture, that of Cheng (1999), Schumacher and Radford (1997) and Radford (1997) confirms this view. Cooke (1999) presents an initial exploration of other ideas. This paper unpacks some of these wider views supported by examples of student work.

3. THE METAPHOR OF PLAY

We use "play" as a metaphor and thus eschew defining it. "Play" gains meaning as a term from the other terms that it entails, the other metaphors that, cloud-like, surround it. Thus, we use a range of terms in conceiving our games. Similarly, we accept a wide variety of scenarios as "games." In the next two sections we describe, first, some of the metaphors play entails and how they relate to learning design and second, some of the games that we have created.

Roger Caillois, (1961) in his classic work, *Man, Play and Games,* describes the universe of play as a cylinder, which is divided into four main *quadrants*, which he calls *agôn, alea, mimicry* and *ilinx. Agôn* is competition, *alea* chance, *mimicry* simulation and *ilinx* vertigo. Games are placed within or between quadrants depending on which of the four principles dominates. The ends of the cylinder correspond to the poles of *paidia* and *ludus. Paidia* implicates turbulence, improvisation, gaiety and fantasy. *Ludus* implicates order, imperatives, rules and the necessity for strategy. *Figure 1* diagrams Caillois's system. His choices of words are practical: he sought the most effective terms and did not limit his search to the French in which he wrote. In doing so, he avoided loading his classification system with unintended meanings. He carries this further in his exposition, choosing different phrases at each explanation of the concepts. In addition to classifying types of play, Caillois also defines play as an activity that is free, separate from reality, uncertain, unproductive (in economic terms), rule-based and make-believe. Like him, we are concerned more with the qualities of play than with the things that set it apart from other activities. Our main concern is to understand how play and designing relate and, instrumentally, to conceive games (play scenarios) that are useful in learning about designing. We use the hermeneutical notion of a metaphor as a relation that simultaneously informs both things being related.

Metaphors put one thing in terms of another. They are ubiquitous in thought.

> *"Metaphor is a tool so ordinary that we use it unconsciously and automatically, with so little effort that we hardly notice it. It is omnipresent: metaphor suffuses our thoughts, no matter what we are thinking about.* (Lakoff and Turner, 1989, p. xi)

One school of philosophical thought, hermeneutics, claims that thought itself is metaphorical.

> *Metaphor ... is no mere figure of speech, but a figure of thought. It is much more than a grammatical form or literary device; it is a cognitive structure which inheres within every transposition of concepts, whether between words or images, a text and its context, the parts and the whole of some gestalt of meaning, or between two networks of statements or two complex conceptual systems.* (Snodgrass and Coyne, 1991, p.8)

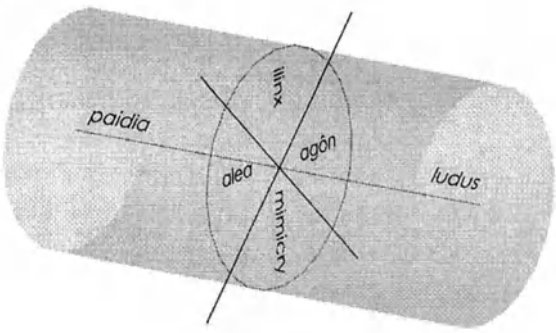


Figure 1. Caillois's classification system: *agôn*, *alea*, *mimicry* and *ilinx* and variations along a spectrum from *paidia* to *ludus*.

In hermeneutics, all utterances can be understood deeply only by considering their metaphorical play with other utterances.

Metaphors thus *entail* other metaphors. It is in through the exploration of entailment that understanding grows. We have found several metaphors particularly useful in understanding the nexus between play and designing.

3.1 Games

A game is a scenario within which play occurs. Games are contingent agreements about how play proceeds. They prescribe the choices and actions that can be made and the effects of these actions on the play of the game. Playing a game can be seen as an exploration of paths of choices and actions

from all those available to the players. A session of design work is game-like in each of the features above. Designers often work within a constrained frame of an existing design and self-selected operations and work out consequences of these over time. Unlike many simple games, designers habitually change their frame and operations over time. In design, the agreement about how to proceed is not fixed.

3.2 Exploration

The so-called exploration metaphor names one of the principal enduring threads of research in design computing. It has many variants, but all share a common abstract symbol level comprising a set of potentially variable generative operators whose action traces out a space of related designs. For most of its proponents, the term "exploration" captures a sense of open-endedness, a contingent sense of both where you are going and what you might find when you get there. Playing a game can be seen in the same light. Playing a game traces out one path through all possible (potentially infinite) playings-out of the game. Taken together, all such paths trace out the space of potential game play. In some games, like Monopoly, the space is fixed—both the allowable moves and the end state of the game are given. In other games, particularly role-playing games, the space is fluid: it can change before or during game play.

In most games play focuses on the present. What is important is the immediate state of the game. The exploration metaphor accesses the previous states of a game as well, making it possible to pick up a game from any previously discovered state of play.

The Virtual Gallery (shown in *Figure 2*) is a generic online gallery system supporting four roles: curators, exhibitors, critics and viewers. It enables a form of exploration through its *response exhibits.* These permit a properly authorised exhibitor to post an exhibit in response to another exhibit. The result of many such postings is a tree of exhibits related to each other by the action of the exhibitors. At first glance, the Virtual Gallery appears to be simple threaded discussion board and it is, when its authoring roles, access, content, critique and survey controls are disabled. Its utility is the ability to structure a particular exhibit to the requirements of a particular learning situation, in essence a designed and structured conversation amongst learners and instructors. This is the argument long put forward for workgroup software—structuring conversation in an organisation is one of the main functions of information technology (Winograd and Flores, 1986, p. 159).

3.3 Balance of forces

In many games progress is easy, but completion is hard. In UNO, a child's card game, the object is to place all of your cards. When you have lots of cards, this is easy. When you have few cards, it is more difficult and other players tend to force you to pick up more cards. Out of this balance between ease and difficulty comes most of the game's intrigue.

Figure 2. An exhibition and exhibit in the Virtual Gallery.

The game entitled *125K Litres of Space* (shown in *Figure 3*) demonstrates forces in balance. The aim of the game is to create two objects that collectively fill and are contained within a five-metre cube and whose volumes sum to 125 cubic metres, (that is, the objects have no intersection). Each object is to be a strong visual composition in its own right. Several different balancing acts are at play. First, the objects are utterly dependent on each other. A compositional change to one relies on a complementary change to the other. The balance trick is to maintain interesting compositional qualities in both objects simultaneously. Second, there are several implicit learning agendas here. Technically, playing the game requires simultaneous attention to the Boolean operators (union, intersection, difference and split), the allocation of objects to layers and to the manipulation of views. Each of these operations is relatively simple in isolation, but their putting together reveals and requires use of higher-level idiom in the particular CAD package used (form•Z). Co-existing with this technical agenda are the compositional issues for each of the objects.

3.4 Tactility

Many games are played with equipment. Chess, checkers, backgammon and go are played on a board. Cricket requires a pitch, stumps, wicket, ball and bat. Sailing needs a boat, water and wind. Froebel blocks are smooth, clear hardwood. In all of these the equipment itself is part of the game. The sound of stones on the go board, the ground and grass qualities of the pitch, the handling of the boat and the touch and weight of blocks all contribute to the pleasure and skill of play. Design is "played" in media and these effect both its process and product. Like learning a game, part of learning to design lies in comfort with the medium. Batting, bowling and fielding skills are the elements from which cricket strategy is formed. Drawing, model-making and computing are the media with which design abilities are learned. Design may be played out in media, but it is realised in wood, concrete, steel and other real materials. Sophisticated materiality is one of the recognised dimensions of mature practice in architecture, yet dealing with materiality is one of the weak points in many, if not most, schools of architecture world-wide. Like play, design is inherently tactile.

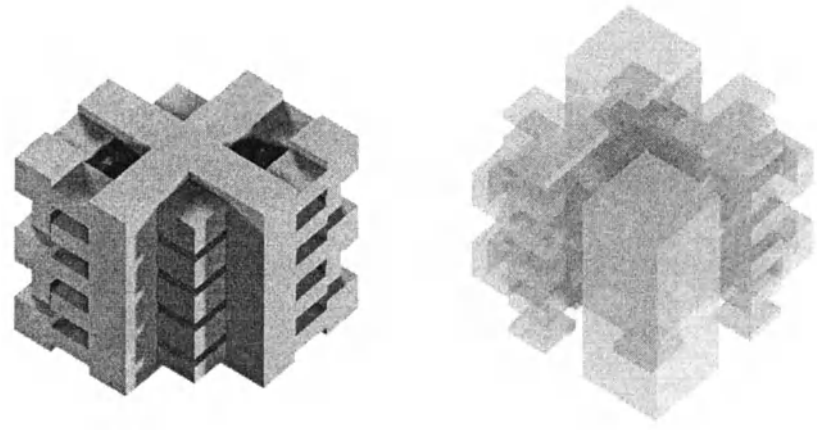


Figure 3. One state of play from the *125K Litres of Space* game.

Tactility is tough with computers. The medium is abstract. There is a sound argument to be made that digital media reinforce an already alarming trend in architectural education: fewer of our students are "makers" and those that have "made" have made fewer things. Yet, if recent national and international design awards, (Anon 1997, 1998, 1999) are any indication, society places considerable value on architects who have profound knowledge of building construction and how it can be used to augment an architectural idea. Developed practitioners of *design making* can relate the

distinct domains of architectural ideas and a constructed reality. Beginning students and other laypersons have particular difficulty with such discourse between a world of ideas and a world of physical form.

Steps can be taken to connect digital media with the objects that they represent and to assist students in becoming poetic design makers if that is their choice. Mitchell's recent championing of CAD/CAM in linking architecture and construction is one such step in professional practice and education. Full-scale computer-controlled building construction demands a new engagement with questions of materiality. The recent construction work on the Expiatory Church of the Sagrada Familia (Antoni Gaudi) is another example of a close connection between digital media and the new forms being created. These efforts though need to be seen more as the beginnings of a new approach to materiality and less as redress for an impoverished culture of learning about the physical in architecture.

We do three main things to help bring students in better connection with the things they are modelling, all of which are demonstrated to some extent in *Figure 4*. The first is simple, yet has wide effects. Our games have carefully crafted surface styles, sometimes realistic, sometimes caricatured. To work with the games is to continually experience one aspect of best practice in representing materiality. The second is to use as case material architecture that is expressive of construction. We have made wide use of the genre lately known as *Australian Lightweight Houses* in the material we give to students. The third is to use digital media to create deeper expositions of construction systems and processes than are generally available to students.

3.5 Intrinsic rewards

Scholars of play such as Bower (Bower, 1974, p8) and Caillois (Caillois, 1961, p6) are emphatic on certain qualities of play. It is its own reward. It is voluntary. Caillois (Caillois, 1961, p6) argues further that these two qualities create a separate world in which play occurs, a world isolated from the constraints and confusions of ordinary life. When either of these two conditions ceases so does play. An athletic scholarship for a high rank in an event, a parent pushing for academic achievement, the social hierarchy on the surf-beach are all things that detract from the intrinsic reward of play. The direct rewards of play are play itself and the successful conclusion of play, however that might be conceived. In architecture, Archea (Archea, 1987) for example, connects to the play metaphor through these qualities of intrinsic delight and separateness from the world. Play and designing are both states of being absorbed in action for its own sake. At first, the intrinsic reward, voluntary participation and separateness required of play seem

antithetical to the idea that play might be harnessed for something so purposeful as learning to design. Can a person be made to play? Does the requirement to play a game in order to learn an idea or skill destroy the illusory world so essential to sustaining interest in a game? The answer to this from the large literature on games in learning appears to be that intrinsic interest is self-fulfilling. If an activity is sufficiently interesting and relevant then learners will enter as they would a voluntary game. In learning, games are about being involved.

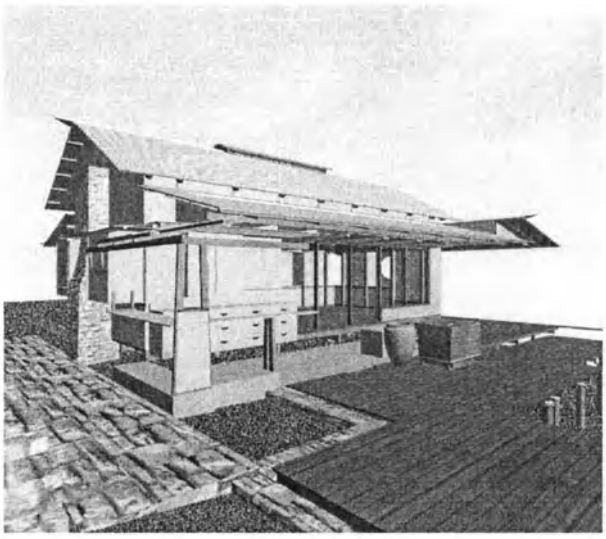


Figure 4. A model of an Australian Lightweight House, the Pittwater House by Richard LePlastrier.

While we aim at creating high levels of intrinsic motivation in all of our games, some are better than others. One of these is *The Nightclub*, a game to design a nightclub within a prescribed envelope (*Figure 5*). The student research for this game is serious fun! The game stresses a presentation along three themes: experience, explicit composition and the linking of these two—exactly the themes of the course it culminates.

3.6 Embodiment

Game play often has an "I," an identity through which the player participates in the game. In Caillois's terms, mimicry is pervasive. In athletics the "I" and the player are one. Even a game so abstract as chess preserves a role for the "I" in the king of each side—the player on whose security the game is won or lost. Designing suggests several "I's." There is

the "I" of the designer creating a work and the "I" of any of several putative users of a proposed building.

The game, *The Children's' Playground* (shown in *Figure 6*) embodies an "I" both familiar and foreign. The aim of the game is to design a playground in two parts having a formal "conversation" with each other. The playground is for three-to-five year olds and is presented largely through animation of a child's experience at play. The "I" is a child, usually the student as a child—from our point it is great fun to watch and listen as students place themselves, both imaginatively and physically in the child's world. In this game the students wrestle with dualities in form-making and the playing out of those dualities in the experience of form through play. Technically, this is their first animation. The task of creating an animation provably taken from a child's perspective is intended as a technical discipline, and, more importantly, a lesson in deliberate communication as part of designing.

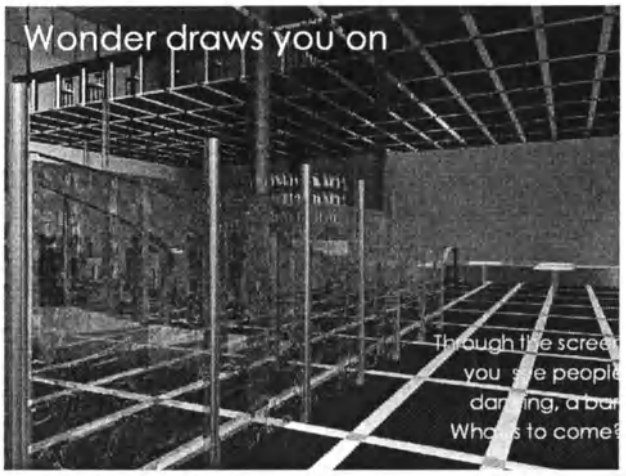


Figure 5. A slide from a student presentation of The Nightclub.

3.7 Rules

Games have rules. Most often these are positive statements about how a game is to be played. In established games the rules are formalised: "a knight can move, ignoring any intervening obstructions, in any L-shaped pattern having one side of two squares and the other of one square." In some games, particularly those of children, rules are often negotiated at the outset: "...let's play go-broke Monopoly" (in which the object of the game is to lose all money and property and most of the rules are reversed)(Woodbury 2001). Caillois points out that even in direct role playing, there are rules—

the rules that govern the role being played (Caillois, 1961, p. 30). Caillois associates rules with the *ludus* end of the *paidia/ludus* continuum in his classification. For Caillois, *ludus* is the order or discipline that develops from the tumultuous and open play of young children. The rules that bound a game give it its intrinsic interest—players seek the skill needed to navigate the landscape of choice created by the rules. The exploration metaphor in design trades heavily on rules as the generative operators for a design space. Like children's' games, the rules in exploration are meant to be broken or replaced by other rules. This feature is taken as definitive, effectively separating exploration from the putatively simpler process of search.

Figure 6. A child's eye view of a Children's' Playground.

We have long traded on the notion of rules in correspondence with the "moves" designers make in their work. We present to students the notion that a few ways of making form (often one!) interpreted and applied throughout a form-making process is an aid to creating spatially and experientially coherent places. Many of our games have relied on rules in one form or another. Though we have used formal grammatical rules directly in our teaching, we have tended to use the metaphor of the rule itself metaphorically—our interpretations for students have stressed effect more than formal machinery. In this genre, games typically provide a site (a game board), a set of physical or spatial components (the game pieces) and some preferred spatial relationships which students use to explore a world of possibility. We encourage particularly those who need extension to break

the rules by changing the site, the pieces or the spatial relations. We have played such rules games in both physical and virtual media.

We show in *Figure 7* one such rule-based game, *Tactile Objects,* which has, for two years, been the first game in the first computing subject taken by our first year students. The game comprises a grid, a collection of related objects and the directions to use the snap options in move commands to relate objects to each other and to have realisable, that is, face-to-face, relations with no spatial overlap. The design aim of the game is to create a composition that would be pleasing to hold in your hands. Technically, this game presents, for most students, the first time they have used the particular CAD system (form•Z) and, for many, the first time with any CAD system. The aims here are to learn to move objects in three-dimensions and to move objects precisely using object snaps. The objects use the square root of two widely in their dimensions. This prevents any possibility of using a grid or direct number entry to locate objects.

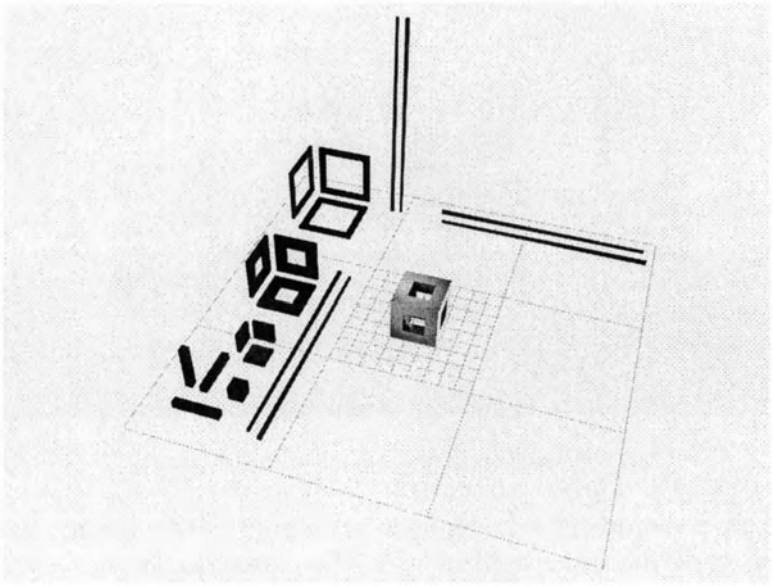


Figure 7. The board for the game Tactile Objects,

4. DOES PLAY "WORK"?

In this paper we have presented some of the ideas pushing along our play with learning. We have not discussed the serious question that any of this actually works in helping students be more confident and competent makers of tactile architecture. As part of our project we are conducting formal

evaluations of our practice of play in learning in early tertiary design education. Our methodology is mixed but trades heavily on the use of trained participant-observers to tease out what actually occurs in the classroom and studio (Shannon, 1995). We can report with some confidence positive effects on student confidence and self-assessed competence with form-making. We can also report some specific changes that we have made based on our findings. Streamed tutorials based on accomplishment in previous courses have allowed us to take advantage of the different learning styles that correlate with differing levels of achievement. Accommodating those learning styles in our instruction created this year the bizarre situation that the group containing the previously under-performing students achieved a higher overall mean score than the previous high performers in our introductory form-making subject. We also know that encouragement of active collaboration in class and evaluation processes can aid better linking of design and its criticism. These and other results from our project in progress await future publication.

5. SUMMARY

The cloud of metaphors around play is much denser that our sketch here describes. In reading the extant literature, examining our existing games and working to construct new ones we have touched on a fraction of the metaphors we suspect are "out there" in the play of conversation. Archea writes of *puzzle-making* in architecture (Archea, 1987) and, though his usage is unconventional, his puzzles are a powerful symbols for events at the core of designing. Effective learning has much to do with taking *risks*, and universities with sheltering students from the effects of intellectual risk-taking. Alexander and other have played out *pattern* as metaphor and a practical tool for thought in design. The dual ideas of *choice* and *constraint* temper much thought about designing. Writers on both games, for example, (Caillois, 1961), and design, for example, (Archea, 1987), point to a protected place at the heart of play. This *world apart* which invokes a *mystery* or *secrecy* about play seems to us both crucial in understanding play and design, and the source of considerable mischief about inherent and mysterious "talent." Though we known of many others, we conclude our short list here with the idea of *winning* and its converse *losing,* arguably the dominant contemporary conception of games, though only one of Caillois's four principal qualities of play. The university experience too is often distorted by this metaphor in the form of competition amongst students. Countervailing metaphors are *collaboration* and *co-operation.* Play upon these can build the culture of peer learning that is the goal of much current

learning theory. Dr. Seuss had it right about the folly hidden in a competitive view of play (Seuss, 1990):

I'm afraid that some *times*
you'll play lonely games too.
Games you can't win
'cause you'll play against you.

6. REFERENCES

Anon, 1999, "Architecture and place: celebrating Richard Leplastrier's RAIA Gold Medal", *Architecture Australia*, 88(1), p. 56-69.

Anon 1998, "Gabriel goes gold: Gabriel Poole's medal-winning career", *Architecture Australia*, 87(1), p. 48-59.

Anon 1997, "Sunshine Coast Library, John Mainwaring and Lawrence Neild, Sir Zelman Cowen Award for Public Buildings", *Architecture Australia*, 86(6), p. 46-47.

Archea, J., 1987, "Puzzle-making: what architects do when no one is looking". In Kalay, Y. (ed), *Principles of Computer-Aided Design. Computability of Design*, John-Wiley & Sons, New York, NY, USA, 1987, p. 37-52.

Caillois, R., 1961, *Man, play and gmes,* The Free Press of Glencoe, Inc., New York, NY, USA.

Cheng, N., 1999, "Playing with digital media: an approach to teaching computer graphics", in Ataman, O. and J. Bermudez (eds.) *Proceedings of ACADIA '99*, p. 96-109.

Cooke, D.M., 1999, *Play/cyberspace/bodily experience,* Advanced Study Report, The University of Adelaide.

Bower, E.M., "Play's the thing", in Shears, L.M. and E.M. Bower. (eds), *Games in Education and Development,* Charles C. Thomas, Springfield, IL, USA, 1974, p. 5-11.

Lakoff, G. and M. Turner, *More than cool reason: a field guide to poetic metaphor*, The University of Chicago Press, Chicago, 1989.

Radford, A.D. (1997). "Games about learning and form in architecture", *Proc. ECAADE'97.* (cdrom; unnumbered pages).

Schumacher, P. and A.D. Radford, (1997). "Games in virtual blockland", in: Yu-Tung Liu, Jin-Yeu Tsou and June-Hao Hou, eds., *CAADRIA'97: Proceedings of the Second Conference on Computer-Aided Design Research in Asia*, Hu's Publisher Inc, Taipei, p. 277-286.

Dr. Seuss, 1990, *Oh, the places you'll go!, Random House, New York, NY, USA.*

Shannon, S.J. 1995, *The studio critique in architectural education*, PhD Thesis, The University of Adelaide, Adelaide, South Australia.

Snodgrass, A. and R.Coyne, 1991, *Models, metaphors and hermeneutics of designing,* Working Paper, Faculty of Architecture, University of Sydney.

Winograd, T., and F. Flores, 1986, *Understanding computers and cognition: A New Foundation for Design,* Addison-Wesley, New York, NY, USA.

Woodbury, C.C., 2001, personal communication.

Exploiting Tools of Evaluation to Improve CAAD Teaching Methods

A Case Study of Inter & Intra ECTM Model

Ra'Ed K. QaQish
University of Petra

Key words: Evaluation, Teaching methods, Interactivity, Effectiveness

Abstract: This paper reports on an ongoing research study model into the Evaluation of CAAD Teaching Methods (ECTM) of which a number of resolutions and strategies were attained via an empirical investigation. The first stage of the study findings proposed a framework for the evaluation of architecture courses in tandem with CAAD. The second stage was based on the Inter & Intra ECTM design model as a strategy for acquiring solutions to CAAD problems through the exploitation of CAAD evaluation tools. The ECTM model structure criteria: the Model Concepts, the Operational Context, Dialectic Meanings, Relational Context, Performing Methods and Level of Integration were illuminated. ECTM model has a twofold involvement junctures, which describe CAAD evaluation behaviour. The first involves the evaluator in an interdepartmental comparison of CAAD integration into the curriculum, and/or between schools of architecture. The second engages the evaluator in an intradepartmental study of CAAD integration, and within the institution. The study projected an attempt to validate the Inter & Intra ECTM design model in concert with evaluation. The paper presents an extended description of the objectives, procedures and testing designed for the two above-mentioned junctures composing the proposed ECTM case studies. Sequences of methods of data collection employed as a vehicle for the ECTM were Kirkpatrick model, questionnaire survey, observation (using an ECTM checklist) and experimental studies. The paper also explores variables and indicators used, and advances to shed some light on the methods of statistical analysis employment. ECTM model as a tool to attain CAAD effectiveness might redefine the role of collaborators/ team partnerships in CAAD tuition; and induce the level of technology selection and adaptation amongst schools, e.g. tutors and coursework interconnectivity. The ECTM model may also work as a framework of strategies to augment interactivity and positive learning amongst both staff and students.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 215–230.
© 2001 *Kluwer Academic Publishers.*

1. ESSENCE OF EVALUATION INSUFFICIENCY

In recent years design schools, and CAAD tutors accustomed themselves to adapt new CAAD teaching methods but small number of those theories have been evaluated or validated. An effective tool to remedy this problem is Evaluation, which many seem to ignore and disregard as neglect. Absence of evaluation influences the efficacy of CAAD in design schools. Evaluation of CAAD instruction is a fundamental approach to address CAAD integration efficiency and problems to achieve effectiveness and productivity. As a result, the ECTM model was instigated as a structured tool of CAAD teaching and learning evaluation at schools of architecture, art and design in which it provides a framework for future self-conducted or organisational evaluation of courseware, software or curricula. The initial work involved the development of a computer-aided design case studies suite to validate such evaluation (QaQish 1997). The study anthology investigated the evaluation of CAAD teaching methods and integration effectiveness of CAAD operational suitability in the design studio. Also, to what affect CAAD influences the design process tuition using the new ECTM model for investigation as a vehicle. Ultimately, this instrument intends to provide an additional interactive learning device, to validate and improve teaching methods amongst tutors. ECTM tools have been designed in the form of inter and intra ECTM model to compare the outcome responses of different groups of students. E.g. those exposed to a CAD package in addition to conventional training, and those receiving only conventional training can now be evaluated in several methods and tools (see Table 3) based upon the confirmed hypotheses that there are fundamental differences between computer assisted tuition and conventional pedagogy (QaQish 1997). Foster and Booth (1993) proposed a twofold approach methodology of CAD evaluation; fundamental approach, by its ability to attain the most wanted learning objectives in terms of individual understanding and satisfaction of use, and overall approach, in terms of awareness of organisational aims and goals, e.g., cost efficiency and standardisation. Such proposal has been incorporated into the methodological design of ECTM model. Methods of data collection and tools of evaluation incorporated are: students profile, pilot studies, observations and questionnaire surveys. ECTM variables vehicle consist of CAAD Tutor, Course Materials & Contents, Class Environment, Use of Media, Delivery Methodologies, Administrative Briefs, and Overall Effectiveness of CAAD event (see section 3.3). Several other cross-sectional variables used as vehicle are the levels of students' performance, attitudes, knowledge, new-stand, creativity and skills.

1.1 CAAD evaluation definitions

Thorpe (1988) defines evaluation as the collection of analysis and interpretation of information about any aspect of a program of education or training as part of a recognized process of judging its effectiveness, its efficiency. Rowntree (1992) argues this definition suggesting that evaluation does not equal assessment, and that evaluation is a planned systematic and open attempt. It ought to be understood that although used interchangeably the difference between CAAD evaluation and CAAD assessment may best lay in the levels of investigation (see Table 2). Successful CAAD evaluation is a collection of distinctive, measurable, and pragmatic program objectives. To verify that CAAD integration has been successful; CAAD evaluators must define measurable objectives, even if effective assignments were delivered. Thus, effective evaluation facilitates useful CAAD course objectives design (planning) and management of achievements (course outcome)- *statistical test to use: apply for measurement a Paired-Samples T-Test procedure (compare the means of two variables for the a number students under two different states: before and after exposure to a CAD experiment or full rendered course)*. The use of this statistical test produces results when one has a small sample size (Clegg 1995). Whereas CAAD Evaluation is related to the macro or holistic level of the CAD learning event (INTER-ECTM), in relationship to the context of learning process and environment and all the factors that go with it. CAAD assessment conversely acts as the measurement of the level of the student learning and as one of the elements that chain with an evaluation process, verifying the micro-level (INTRA-ECTM).

1.2 CAAD evaluation methodology

Two distinct methods are documented for CAAD evaluation, the scientific method and the illuminative method. The measurement of the effects of specific variables against the resultant outcomes governs the scientific method. The scientific method examines the achievement of set of objectives in relation to a learner's pre-knowledge and skills, e.g. *how would you rate your gained knowledge and competence in each of the following areas of specific CAAD software?* It also engages in the measurement of the efficiency and effectiveness of the educational intervention and its outcomes, e.g. *please indicate the extent that CAAD application has in improving the over all effectiveness of the following criteria in the design studio?* In contrast, the illuminative method examines and investigates the process of the CAAD interventions on the overall course or curriculum governed by but are more qualitative, and subjective techniques, e.g. *please*

indicate to what extent would you agree with the following statements in relation to your design and CAAD events?

1.3 Reasoning and suggestions

CAAD evaluation should always be a comprehensible and clearly planned task. Both scientific and illuminative methods are used stand alone or combined in reference to CAAD course or environment complexity (General, Specific or In-depth methods in ECTM model). The level of combination is determined by time-scale and evaluation process set by the evaluator and the instigator of the evaluation. Good preparation and pre-planning should aid an evaluation case study to advance towards easier analysis and concrete recommendations; this includes defining investigation criteria. CAAD evaluation should become stimulant to improve a process of innovation within a curriculum together with factors such as CAAD tutor, students, material, and learning environment. Learning materials of any CAAD course depended mostly on refinement by tutors to comply with specific learning objectives upon which leaves an impact on the learner by the extent of its achievement level (scientific approach). In design programs with CAAD interventions, the need of validation supports the evidence after reasons to conduct evaluations, including the following factors (QaQish 1997):

1. Furnish statistics about value delivery that will be effective in CAAD course, see examples of different questions in Table 5. ECTM model capitalised on the significance of the concepts of Likert rating scale for measurement expressed in units using a triangulated framework of measurement (nominal, ordinal, interval) (See Tables 1, 3, 4).

Table 1. A suggested measuring Likert five point scale to test significance of CAAD.

5	4	3	2	1
Of great importance	Important	Of some importance	Unimportant	Not sure
Very Confident	Confident	Some Confidence	Little Confidence	No Confidence
Excellent	Good	Fair	Poor	Not Sure
To great extent	To good extent	To some extent	To no extent	Not Sure

1. Decide the level of appropriateness and effectiveness of CAAD course for students, e.g. Please indicate how important the following criteria have been for you to determine an over all satisfaction with the effectiveness of the CAAD? (See Table 1 for scale measurement)
2. Determine whether programs "make a difference".
3. Provide the means for CAAD staff to prompt effective changes.

4. Advance CAAD susceptibility.
5. Provide CAAD staff with information to improve service delivery.
6. Explore CAAD effectiveness in practice amongst graduates.
7. Report program objectives have been successful or not.

Evaluation mechanisms for design schools can be tailored to CAAD programs' objectives and necessities, e.g., *schools with inadequate evaluation resources would concentrate on detecting how effectively CAAD courses are delivered- use internal questionnaire survey technique.* Schools with limitations on CAAD programs could identify how those limitations and constraints affect program delivery, e.g. *budget, CAAD staff availability and competency, space availability, curriculum flexibility, and course structure.* CAAD staff with evaluation capabilities and well-built resources can employ evaluation to study how successfully they are affecting student behaviour, performance, skills, knowledge, and use the results to enhance projects' brief. Design schools carrying out evaluation are responsible to report on the impact of their CAAD programs on student behaviour and performance.

2. CAAD COURSEWARE EVALUATION

Laurillard (1993) reiterates the importance of the evaluation process undertaking at every stage in the design, production, and integration of a new educational intervention whether courseware is a complete course, part of a course, or a particular session or teaching aid are all treated as equal in the process of evaluation. Teymur (1992) suggested several criteria form courseware evaluation, objectives and aspects, medium, methods. Hennessy (1982) maintains that to use a curriculum's development method effectively an established performance achievement is the basic design objectives. So, specific performance standards are needed and preferred. Thus, courseware evaluation could permit CAD tutors to examine other factors than software evaluation such as aspects like productivity, user-friendliness, navigation, screen design, text layout... etc, leading to the successful integration of the CAAD course into the course itself or the curriculum. Courseware factors are namely: educational setting (Environment, facilities, staff, materials, courseware and software, and administration), aims and objectives of the CAAD course, CAAD teaching approach, CAAD learning strategies, CAAD teaching and assessment methods, and implementation of CAAD strategies. The most important objective behind using the above factors for evaluation is to support creation of a new learning experience both more efficient and effective to constantly recuperate CAAD teaching difficulties. On the other hand, evaluation is a time consuming application and effective planning can

reduce time and effort in the tangible evaluation and analysis process. Small evaluation schemes are advised in favour of a large evaluation ones. While, Romiszowski (1988) makes a distinctive analytical separation between scope of evaluation levels and depth of evaluation levels (Level 1: project level, Level 2: curriculum level, Level 3: unit level, Level 4: learning step level). The ECTM model perceives this in a rather more practical approach assigning specific inter and intra levels (see Table 2). It is important to predetermine the induction of CAAD evaluation levels at each CAAD teaching and learning stage. This works interchangeably with levels of implementations in the ECTM techniques. When looking at the behavior of ECTM, architecture and design schools could view the evaluations analysis and its process to be adequate for CAAD course assessment. CAAD evaluation determines whether delivering CAAD is being carried out in an appropriate and efficient manner to guarantee sets of goals are being met and to improve and maintain efficiency in CAAD. CAAD evaluation also determines whether CAAD programs are effective in making a difference for new graduates. Thus a post-evaluation program of graduates group is supplementary requirement to in-school evaluation. This paves ground floor to examine interventions changes into students' behavior or design studio approach to enhance CAAD program or validate the curricula need changes. Schools and educators must recognize the need for:

1. Evaluate the structure of the curriculum to successfully add CAAD.
2. Evaluate transformation of course objectives and assessments in the design studio with CAAD.
3. Evaluate CAAD teachers/tutors while delivering comprehensive CAAD.

Effective and efficient changes in the design studio arise when competent students and confident tutors are resourceful, appropriate CAAD applications are available and properly delivered through a well-established CAAD- design studio program. Evaluating CAAD, learning and teaching outcomes could identify changes that may have occurred in architectural or design tuition because of CAAD integration and implementation. Analysing these changes may help to determine whether there are characteristic and behavioural changes of CAAD program, which indicate changes occurrence in program activities. Indicative of such a direct change in behaviour transpire because of a CAAD program is commonly difficult, because students' behaviours towards CAAD software are expected to change over time as CAAD packages are constantly and rapidly advancing. E.g. *one of the problems of integrating CAAD in the design studio is the teacher's lack of competence, and method of instruction. CAAD problems may lie in the organization and management of teaching and the improper course objectives.* Seven instruments are strategically essential as a pre-evaluation operational tool for carrying out CAAD evaluation: costs of an innovation (Courseware and Software), development needed in advancing and

improving staff and students, measure efficiency and increased competence against improved benefits in the institution, measure effectiveness and suitability of learning events, against aptitude of the institution, numerals of budget and students before or after the department introduced CAAD courses, achieve enhancement of concepts represent high-quality success and aspiration creativity and cost and expenditure validation and sustainability.

3. THE INTER AND INTRA ECTM MODEL

Autonomous response built on rational and empirical analysis to the real problems of CAAD teaching & learning has not been implemented during either the design studio and core courses or impartially the computer laboratories. At hand, sufficient indications suggest potential insufficiency on methods to validate CAAD. The evidence suggests that CAAD teaching & learning are confronting problems when administered and instructed in design studios. Thus the aim of the ECTM model is set to determine the most effective standalone or combinations of teaching and learning strategies that may be employed in the decision-making teaching and tutoring of CAD courses and students by adapting a series of CAAD evaluation techniques (see Table 2).

Table 2. Inter & Intra ECTM model structural design

Dialectic meanings	Performing methodology and level of integration			Criteria of study
		(Relational context)		
Inter-Departmental Evaluation in Depth of CAAD Integration	Experimental Quantitative Study	IN- DEPTH Experimental Survey (Conventional + CAD)	Appraisal/ CAAD Integration Appraisal/ Teaching Methods	Measurements Projects Subjects Objectives Course materials Computer labs
			Themes:	Criteria
Intra-Departmental Comparison of	Questionnaire Quantitative	GENERAL Wide Ranging Survey (No. Schools)	Labs Dept. Policies General Info	Physical Facilities Faculty Qualification

Curriculum and CAAD integration	Study	SPECIFIC Emphasis Survey (One School)	Integration Administration	Program Goals Structure of Curriculum

This research previously investigated four principle problems in conjunction with the establishment of the ECTM model toward the solution of CAAD validation:

1. Evaluating CAAD substantiation in the architectural curriculum and the methods and concepts of CAAD integration in the teaching of architecture.
2. Evaluating the teaching methods in terms of effectiveness and efficiency.
3. Evaluating CAAD integration in terms of its effectiveness and appropriateness of use in the design studio.
4. Evaluating the use of CAL in CAAD, the different types of approach to instructions, the instructional strategies that can be employed in computing teaching when integrated with design.

Kirkpatrick's model (1977) was used to assist ECTM. Kirkpatrick argues that an innovation in any institution will unquestionably have an effect on the institution, but will not radically change its structure. Eraut (1969) identified three stages of curriculum development in the new area of technological education: 1) Formulate aims, 2) Develop a mechanism of achieving these objectives, 3) Select the objectives according to the aims.

3.1 ECTM model factors

The TLTP initiative context has two main interests for evaluation both effectiveness and efficiency built on the need to justify technology-based learning. Four other factors could be integrated with those two, these are relevance, performance, timescale and creativity in relationship to the applicability and appropriateness to the intended employers and users of the technology, the teachers and students, even the department or institution. To evaluate CAAD, ECTM factors must be appropriately used to define creativity and performance of the students' learning events and the teacher competency. Attaining the objectives of CAAD in the learning event would help to assimilate knowledge, skills, and attitudes. Two ECTM factors that would contribute to achieving effectiveness are recognized as: Cognitive: Knowledge. Psychomotor: Skills. Whereas, three ECTM factors that contribute to achieving relevance and outcome are Affective: Attitude. Creativity: Innovative. Performance: Achievement measures (QaQish 1998).

3.1.1 Effectiveness factors

- The Cognitive Factor: (assimilating Knowledge). Bloom, Madaus and Hastings (1981) system requires a learner to demonstrate an increased level of processing knowledge: or cognitive skills. It tests the leaner handling of knowledge, and compares different methods of evaluation in any learning event.

  The Affective Factor: (attitudes) Krathwohl, Bloom and Masia (1964) suggest that the learner should know the appropriate attitudes and suitably demonstrate the event.

3.1.2 Relevance and outcome factors

The Psychomotor Factor: (Skill manipulation) Sax (1980) focuses on skills associated with dexterity (proficiency, ability, manual skill), hand/eye co-ordination and error reduction in the human use of devices. It helps in determining whether the low level of skill demonstrated by students is due to a lack of knowledge, or attitudes.

The Creativity Factor: Creativity is the vehicle most design studios use to transfer knowledge, skills, and attitudes to the students and expressed in their project's scheme.

The Performance Factor: Achievement measures. Mager and Pipe (1981) argue that performance level reached amongst students and teachers typically indicate the standards settings in education.

3.2 ECTM activities initiation

Through ECTM activities, CAAD tutors, or administration decide what CAAD courses to offer and how appropriate they are carried out. The appropriate evaluation activities to be inaugurated by ECTM model are:

1. Establish the value and use wanted when integrating CAAD, e.g., *determine what attitudes, knowledge, skills, or behaviours' changes CAAD course poises.*
2. Follow CAAD program objectives, e.g., *design a framework that indicates students' productiveness, how much CAAD knowledge is delivered, how students rate the CAAD knowledge they receive, and which CAAD strategies are most promptly adopted by CAAD staff.*
3. Choose from CAAD alternative program approaches already undertaken, e.g. *comparative analysis of curricula or CAAD strategies to determine which ones accomplish the goals.*

4. Experiment and appraise new CAAD program designs to determine the extent to which a specific proposal is being implemented carefully by architecture school or the extent to which it interests or engages students.
5. Establish CAAD course objectives and determine the specific indicators for testing, e.g. *the tangible performance, skills, knowledge, attitudes, or behaviour measures. This will illustrate the level and extent of success in CAAD objective effectiveness and appropriateness.*

3.3 ECTM b/variables

ECTM variables work as a vehicle to conduct CAAD evaluation and to help bridge the problems of good CAAD integration within the design studio.

1. CAAD Tutor's (Louden 1991) ability indicators are: to deliver, explain, interpret, use, design, adjust to differences in the learning styles, observe classroom pasterns, produce different methods of teaching, and interact with learners.
2. The Course Materials (The Course Contents) (Misanchuk1992) indicators are: Classroom Handouts, laboratory Manuals, Textbooks, Individualized instruction-Packages, Assigned Projects, Tutorial Written Guide, Lectures, Time, Test, Exams and Quizzes.
3. The Classroom Environment (Marsh, 1973) indicators are: Facilities, Class size: Physical size, Class Layout, Control of Seating, Class Light and Temperature, Class Accessibility, Class Furniture, finishes, Class availability.
4. The Use of Media (Slaughter 1990) areas are: CAAD components and properties, and the actual media tutors use to deliver materials. The indicators are Computers- Screen, Mouse, keyboard and PC Case, Peripherals- Plotters, Printers, Scanners, Digital Camera. Software: Applications packages and the use of Multi-Media. Media used in administrating the learning event: Overhead Projector, Slides, and videocassettes.
5. The Delivery Methodologies (Dick & Reiser 1989) indicators are: organization of the learning events, amount of material covered during the learning events, time allocated during various parts of the events, mix of theory and practice, assimilating skills objectives and goals of course.
6. The Administrative Briefs. (Rogers 1983) indicators are: availability of information to the learner, availability of the facilities for the learner, gaining information about the course, communication between the learner and teacher or administration.
7. The Overall Effectiveness of the event (Bloom, Madaus & Hastings 1981) is concerned with the applicability of the learner objectives and goals. The indicators are: Developing Skills, Gaining the knowledge of

the concepts and principles of the CAAD. Develop the attitudes necessary to achieve the directed goals (Standpoint of view, Viewpoint, and New aspects).

3.3.1 ECTM methodology

Tables 3 and 4 show the ECTM methodology in which a suggestion of a framework of evaluating CAAD integration via a series of experiments in the form of fieldwork observations and questionnaire surveys is proposed. It also furnished an empirical investigation into the functional and theoretical usage of CAAD in schools of architecture via a suggested scheme of case studies (see sample questions, Table 5).

Table 3. Levels of ECTM model evaluation techniques

Level	Instrument	Design Considerations Next to Level of Implementation
LEVEL 1	Student Profiles Questionnaires	Obtain a profile to monitor changes in both attitude and opinions. Address learner's attitude towards computer-aided design as well as the use of computers in general. Address learner's academic background in the particular CAD area. Address learner's level of knowledge of CAD and Architecture, Design or graphic Design.
LEVEL 2	Global Questionnaires: Permissibility of a staged Pre and Post Tests	Proper layout and ample space for response. Reduce open-ended questions. Minimum number of questions Use standard Likert five point scale (see Table 1) Anonymous Pilot questionnaire
LEVEL 3	Semi-structured Interviews	Administer after questionnaire collection and analysis. Covers missing points in questionnaire. Initiate questions by interviewees. A focused interview diminishes overloaded data collection. Permit group interviews.
LEVEL 4	Observations	Structured observation sheet. Code or shorthand. Sense of required observe patterns. Video-taped recordings Tape-recorder synchronize students' verbal comments and interactions

Table 4. Levels of ECTM model evaluation in relation to techniques benefits and hindrances

Level	Instrument	Benefits	Hindrances
LEVEL 1	Student Profiles Questionnaires	Useful for group of stratified random sampling. Useful to test abilities in a certain group.	Virtually none
LEVEL 2	Global	Massive Info collection.	Low response rate.

Level	Instrument	Benefits	Hindrances
	Questionnaires	Simple, collection, administration & analysis.	Difficult categorization with open-end question.
LEVEL 3	Interviews	Tolerate extended answers. Instructive uncovered points.	Acquire time to arrange conduct. Surplus unwanted data.
LEVEL 4	Observations	Developed and focused sheets ascertain useful data collection.	Unexpected Occurred Events.

3.3.2 ECTM statistical tests

Part of the evaluation process is to produce the proper questions and analyse the returns from the students who participated in the questionnaire surveys, observations and checklist prepared and carried out by the evaluator. Several types of statistical tests, such as the Chi-square X^2, Spearman's *rho*, *t*-tests, and Kruskal-Wallis one-way analysis of variance by ranks can be used. Frequency tables, charts such as bar, line and pie charts can also be used to describe the findings of CAAD evaluation.

3.3.3 ECTM sample questions, applied statistical tests and analysis

Table 5. Sample questions, statistical tests and analysis

Proposed questions	Statistical tests	What is to be expected from analysis? (Objectives)
1. How would you rate the following tutor's abilities during CAAD events?	Use count and rate to indicate level and extents, use one sample –test. Scale of 1= poor, 2= fair, 3= good, 4= very, 5=excellent	• Examine the impact (significant or weak) of the tutor's abilities, management and organisation on the students' performance, attitude and skills. • Relationship/ Confirm or refute the hypothesis that the Students Confidence depend on the tutor's abilities (competence)

A one-sample *t*-test examines whether the tutor's abilities, management and organization mean of distribution differ significantly from the value of 5= excellent which describes a high level of tutor's ability effectiveness. E.g. *hypothesis*: incompetent teacher has contributed to unsuccessful CAD employment, *Results from analysis*: one-sample *t*-test analysis indicates that the mean percent for the tutors abilities (adjust, deliver...etc.), management and organization, were negatively higher at α =. 001 (values of 5=excellent, under 95% confidence). The null hypothesis was confirmed.

Proposed questions	Statistical tests	What is to be expected from analysis? (Objectives)
2. How would you rate the effectiveness of the tutor's methods of instruction during the computer assisted learning classroom events in the following areas?	Use correlation coefficient (Spearman's *rho*) and X^2 tests to test significance and to show relationships, if any between these variables in a tabular format.	• Examine type of association (positive- negative) between the tutor's methods of instruction or tutor's competence and the students' attitudes, performance, knowledge and skills or level of overall effectiveness. • Examine type of association (positive- negative) between the tutor's competence and his/her strengths in addressing new areas of CAAD.

E.g. hypothesis: if the teacher is competent he should be able to improve the student's knowledge, skill, attitude, creativity, new stand and performance in CAD course. Thus the null hypothesis assumes that the teacher has no effect on these student's domains.

Please indicate how important the following criteria have been for a studenst to determine an overall satisfaction with the effectiveness of the computer learning events?	Use Paired Sample T-Test for related samples. Testing a null hypothesis: e.g. there is no behavioural differences amongst the a number of students who completed the CAD design work shop	• Examine the impact (significant or weak) of the learning environment on the overall effectiveness of the CAAD courses. Few areas of investigations are: free time lab indicating and training hours, lab facilities, the accessibility and availability of information and communication with the staff and the administration.

The paired samples statistics table displays descriptive statistics for the test variables, and followed by the correlation table, which displays the relationship between the paired differences with a 95% confidence interval of the difference of the means.

Please indicate the extent of CAAD effectiveness in the following criteria in the design studio?	Use X^2 (Pearson's r) tests using the SPSS package or Minitab. Results of a *chi*-square test indicate whether or not a variable was found to be significant at the level of 0.05 in other tested variables.	• Examine the overall satisfaction of the CAD sessions both regarding the tutors and the school in providing the appropriate information with the administration of CAAD events.

Proposed questions	Statistical tests	What is to be expected from analysis? (Objectives)
Examine the significance of CAAD in your own set of design studios criteria, some of which are production of generation modelling as in parallel drawings, animation or perspectives. Examine the significance of CAAD effectiveness in design studios to promote aspects such as skills, attitude, creativity and performance.		
In one Sample T-test, critical values (Obtained from a number of previously conducted research) of t must be equal to or more than the tabulated value (obtained from t-test table, Clegg 1995 and used by the researcher), to be significant under $p<0.001$. For $df=30$, the tabulated values read 1.697, 2.042, 2.457, 2.750 and 3.646 under level of significance for the two -tailed test respectively 0.10, 0.05, 0.02, 0.01 and 0.001. 95% CI. Please note that Confidence interval (CI) is the range of values within which the means of tutor's abilities statistical results are likely to fall, thus when a 95% confidence interval for the mean indicates that there is a 95% chance that the true tutor's abilities, management and organisation fall within the range of value 5= excellent. This means that the researcher had anticipated that the competence of the tutors is likely to fall way below the excellent range. Other researchers could use their own hypotheses to confirm or refute.		

3.3.4 Outcome evaluation samples from a case study

Students' responses indicated that they used discrepantly conventional drawings, CAD applications or both (mixed methods) during selected design stages. The results showed evidence of a significant use of the mixed method over the CAD alone method. When compared with the conventional drawings, the mixed methods were found relatively similar; there was a trend towards a mix use of methods, developing in the design studio. CAD has significantly affected a number of design areas such as production, scheme & detailed design. Detailed design stage was mostly affected and had the highest response rate from students (14.3%). The least effected stage was the outline proposal. Sketching was found to be the least influenced by CAD course (mean of 1.9 which equals the value no extent), followed by a similar impact on the analytical diagram. The most significant impact was found in the production of perspectives and 3D modelling (mean of 3.2 which equals the value of good extent) (see Table 1 for measuring scale). The tutor's competence may also be related to his strengths in addressing and presenting new areas of CAD, which had ultimately increased the overall effectiveness of the students' attitude, performance and skills. *t*-test result implies (*t*-test =-10.5, $df=34$, the tabulated value = 1.697 $p = 0.05$ for one-tailed test) that the mean for the lectures/tutorials and short briefs were negatively insignificant. Such results could indicate that the null hypothesis was not refuted and the learning materials failed to ensure successful and effective CAD teaching. The learning environment was found to have

influenced the overall effectiveness of the CAD courses, through its impact on the areas of skills, performance, knowledge, creativity, and attitude. There were significant associations between the overall attitude, and the performance level. Schools should consider changes to the design and organisation of the learning environment (computer labs or CAD labs). This will improve the overall effectiveness and thus improve the performance, and attitude of the students towards CAD and the design studio. The administration of the school and the tutor was found to have significant impact on the students' performance and attitudes.

4. CONCLUSION

The paper presented a brief description of the ECTM model, a background on evaluation and its definitions, involvement of evaluation into CAAD. A full comprehension and clear perception of the theory and background to CAAD evaluation is presented in order to achieve enhanced planning, designing and conducting an evaluation program and benefit from its analysis. This paper also suggested means for CAAD staff and architectural schools to carry out necessary evaluations under which appropriate settings they find necessary. However, its suitability is not restrained to CAAD programs since the paper illustrated what purpose and reason CAAD program need evaluation.

The main issue ECTM model addressed was evaluating unambiguous designs to be conducted without academic training in CAAD program evaluation, although training CAAD staff on evaluation is recommended for CAAD teaching and should become an integral part of CAAD administration. An evaluation is a significant tool in developing the characteristic of CAAD course once it is integrated into the fabric of an architectural program rather than advocated after its implementation. CAAD tutors are encouraged to use the results of an evaluation to determine future extent of CAAD involvement and to upgrade the level of CAAD implementation.

4.1 Acknowledgment

The researcher acknowledges and extends thanks to Teaching with Independent Learning Technologies (TILT) Project group for their extensive and constructive research on evaluation at University of Glasgow. http://www.elec.gla.ac.uk/TILT/TILT.html

5. REFERENCES

Bloom B., G. Madaus and J. Hastings, 1981, *Evaluation to improve learning,* McGraw-Hill, New York.

Breakwell, G. and L. Millward,. 1995, *Basic evaluation methods*, BPS Books, Leicester.

Dick W. and R. Reiser, 1989, *Planning effective instruction,* Allyn & Bacon, London.

Eraut M., 1969, "The impact of educational technology on curriculum development", *Curriculum Innovation in Practice*, Edge Hill College of Education.

Foster, J. & J. Booth, 1993, *Evaluating CAL. CiP 93 Conference Proceedings*. CTI Centre for Psychology, University of York, 18-19.

Hennessy, K., 1982, *A system approach to curriculum development, the Manchester project for computer studies in schools, Microcomputers in Education,* Ellis Horwood Limited, Chichester.

Kirkpatrick D., 1977, "Evaluating training programs: evidence vs. proof", *Training, and Development Journal.*

Krathwohl D., B. Bloom and, B. Masia, 1956, *Taxonomy of educational objectives, book 1, Cognitive Domain*, Longman, London.

Laurillard, D. ,1993, *Rethinking university teaching*, Routledge, London.

Louden W., 1991, *Understanding teaching, teacher,* College Press, New York.

LTDI, 1999, *Implementing Learning Technology*, Edited by Greg Stoner, http://www.icbl.hw.ac.

Mager R. and P. Pipe, 1981, *Analysing performance problems*, Pitman Learning Inc., Belmont, California.

Marsh L., 1973, *Being a teacher,* A&C Black LTD, London.

Misanchuk E., 1992, *Preparing Instructional Text*, Educational Technology Publications New Jersey.

QaQish R., 1997, "CAL in CAAD: Inter & Intra departmental computer management learning (CML) in architectural education (AE)". *Ph.D. Thesis*, MSA, University of Glasgow.

QaQish R., 1998, "The effectiveness of CAAD staff, learning environment and CAL materials evaluation", *ELT 98: Innovation in the Evaluation of Learning Technologies Conference*, The University of North London, UK.

Rogers C., 1983, *Freedom to learn for the 80s.* Charles E. Merrill Publishing Company, London.

Romiszowski, A.J., 1988, *The Selection and use of instructional media*, Kogan Page, London.

Rowntree, D, 1992, *Educational technology in curriculum development*, Athenaeum Press Ltd, Newcastle upon Tyne.

Sax G., 1980, *Principles of educational and psychological measurement and evaluation*, Wadsworth Publishing Company, Belmont, California.

Slaughter T., 1990, *Teaching with media: aguide to selection and use*, the centre for the study of higher Education, University of Melbourne, Victoria.

Teymur N., 1992, *Architectural education: issues in educational practice and policy*, Question Press, London.

Thorpe M., 1988, *Handbook of education technology, Percival* and Race, Ellington.

Creating Place in the Virtual Design Studio

Peter Russell
Institute for Industrial Building Production, University of Karlsruhe

Key words: Virtual Environments, Virtual Design Studio, Internet Utilisation

Abstract: The current wave of attempts to create virtual design studios has demonstrated a wide range of didactical as well as computational models. Through work performed over the past year, an evolution of many of these concepts has been created which fosters a sense of place. This aspect of place has to do with identity and community rather than with form and space. Initial virtual design studio projects were often merely a digital pin-up board, which enabled distributed and asynchronous criticism and review. However, the web sites were more analogous to a directory than to the studio setting of an upper level design problem. The establishment of a truly distributed design studio in the past year, which involved design teams spread over three universities (not parallel to one another) led to the need for an independent place to share and discuss the student's work. Previous virtual design studios have also established web sites with communication facilities, but one was always alone with the information. In order to enhance this virtual design studio and to give it a sense of place, a studio platform that serves as a console for participants was developed. The console is a front end to a dynamic database which mediates information about the participants, their work, timetables and changes to the dynamic community. Through a logon mechanism, the presence of members is traceable and displayed. When a member logs onto the console, other members currently online are displayed to the participant. An online embedded talk function allows informal impromptu discussions to occur at a mouseclick, thus imitating ways similar to the traditional design studio setting. Personal profiles and consultation scheduling constitute the core services available. Use of the platform has proven to be well above expected levels. The students often used the platform as a meeting place to see what was going on and to co-ordinate further discussions using other forums (videoconferences, irc chats or simple telephone conversations. Surveys taken at the end of the semester show a strong affinity for the platform concept in conjunction with a general frustration in pursuing collaboration with low bandwidth communication channels.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 231–242.
© 2001 *Kluwer Academic Publishers.*

1. CREATING THE VIRTUAL DESIGN STUDIO

Since 1997, the Institute for Industrial Building Production (ifib) has conducted its upper level design studios in the "*Netzentwurf*" (translation: net design) setting. This essentially involves the incorporation of the World Wide Web into the design studio criteria, particularly in the aspect of presentation. The methods and results of this studio setting have been documented (Forgber and Russell 1999) and evaluated (Russell, Kohler et al 1999) with some clear deficiencies becoming apparent. Work carried out in the summer of 2000 proposed an evolution of the Netzentwurf setting with the creation of a console or platform, which served as the virtual "place" for the design studio.

In general, the members of the institute believe the boundaries of the traditional University are being blurred in their physical, temporal and institutional forms. The tendency to life long learning and the fluidity that technologies such as the Internet offer allow students to continue their studies long after they have ceased to be officially registered as students. In the case of design, the methods for working are significantly different than those of more standard computer based training. The collaborative aspect behoves a different kind of learning environment. The work carried out at ifib attempts to produce web-based design studio environments.

1.1 Web Pin-Ups

The participants in the Netzentwurf receive initial instruction in HTML and are given a blank slate. That is, it is left to them as to how to present their work. The Institute's web site serves as a directory of student web sites. The requirement to display their work in the World Wide Web is a double-edged sword for many of the students. The relatively limited display area poses a challenge as they are used to having tens of square meters of pin up space available simultaneously. Another challenge is the nature of viewing their web pages. Students often create two parallel narratives on their web sites: One for individual browsing augmented by textual explanation and another for public presentations usually augmented by the author's commentary in person.

The students are also encouraged to use other channels for communication such as email, Newsgroups and chats. However, in cases where the studio involved just one institute, the students were all local to the campus and so chose to communicate mostly through face to face discussions. The reasons for this have implications for interuniversity design studios that are not easily answered with more technology. The use of any new technology must have a meaning or use that goes beyond its newness if

it is to have longevity in the design studio. This became clear after the second or third cycle of the Netzentwurf concept where the work of the students seemed to plateau at a certain level of graphical detail. Further to the development of the Netzentwurf was that the separate aspects of the web based design studio took place with separate computer programs and thus, the identity of the Netzentwurf itself became diffuse. (See Figure 1.)

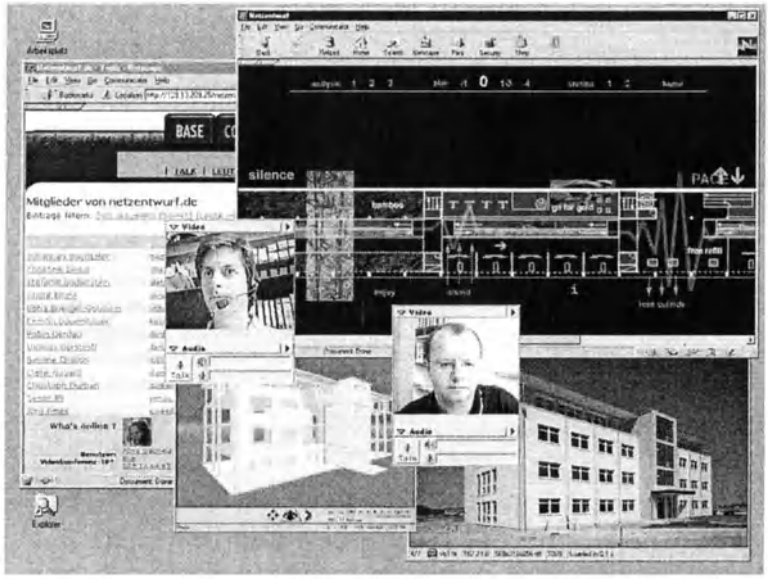


Figure 1: Early Netzentwurf Environment

1.2 Expansion

In the winter of 1999-2000, the Institute carried out an international design studio under the auspices of the Virtual Upperrhine University of Architecture (VuuA). Approximately 150 students from 5 schools in four disciplines (Landscape Architecture, Interior Design, Urban Planning and Architecture) from Switzerland, Germany and France took part in the programme which involved the redesign/reuse of an old Fort outside of Strasbourg.

The sheer number of students, tutors and institutions involved led to a rethink of how the Netzentwurf site itself was organised. Previous studios had involved between 10 and 25 students and so the information, (HTML pages) was crafted by hand. With the VuuA programme, it became necessary to automate the web page creation and maintenance. A system consisting of Active Server Pages (ASP) connected by SQL instructions to a database

allowed the system to be more or less self-controlled. Students could then update their personal information and links to their projects themselves. The use of the database also allowed the introduction of two other innovations into the Netzentwurf setting: the logbook and the competency listings (Koch and Russell, 2000).

Despite the advantages the ASP based web site gave to the students, the potential for cross-pollination and collaboration was barely utilised. The newsgroup discussions and emails exchanges were barely enough to consider the work as being collaborative at all. In post-studio evaluations, (Elger and Russell, 2000) a common theme among the students was that they were always alone with the information. That is, the directory of student work was well linked, but that the work was in some way faceless and that one was never "with" another student to discuss the work. The availability of email and Newsgroups or even chatrooms was not enough to sustain any kind of spontaneous discussion. This is in sharp contrast to the physical design studio where spontaneous discussions are the norm and indeed, could be considered as essential to the design studio experience.

1.3 Evolution

In the summer of 2000, ifib undertook to establish an interuniversity design studio with the Institute for Computer Supported Planning at the University of Kaiserslautern and the Institute for Architectural Design and Information at the University of Cottbus involving several design themes simultaneously. This meant that students from one University could partake in the studio of another University and receive credit without the bureaucracy indigenous to programmes like ERASAMUS. This was based on concurrent research work at ifib (Russell and Forgber, 2000).

Three design topics were offered to approximately 50 students at the three universities with the goal of creating design teams spanning the three universities. The results of the previous semester's evaluations led the tutors to the conclusion that the success of the interuniversity co-operation would rely on being able to create a web based studio setting that would encourage and enable impromptu communication. Work carried out by Claus-Jürgen Schink as part of his Diploma at ifib gave some clues as to how this might be done and were incorporated into the Netzentwurf platform.

The platform was intended as an augmentation to the design studio setting, which would enable geographic separation to play less of a role in establishing student design teams. The organisers also placed a large importance on the participants having previous personal contact prior to the use of the platform. Thus, each design theme started with a 2-3 day workshop where the students were given various tasks in different

constellations in order that they could get to know one another before forming groups and using the platform. This allowed the participants to put not only a name to a face, but also a personality (or point of view) to the said name and face.

2. CREATING PLACE

The goal of the Netzentwurf evolution was to create a sense of place in the virtual design studio. It was not the intention to replicate a three dimensional world where avatars could meet. Cyberspace worlds such as these imply a large amount of computer power and bandwidth and what is more, this vicarious way of establishing place always seems to carry a sense of otherness about it. That is, the three-dimensional worlds such as those created in VRML are often "there" and not "here" for a participant.

In creating place in the Netzentwurf, it was the intention to create the kind of place one experiences when conversing on the telephone. The other person is really not anywhere. The voice is not in the head, not in the phone and not perceived as being hundreds of kilometres elsewhere. It is simply there. However, this is not important so long as the voice comes through which thereby establishes the presence of the other person. The participants in the Netzentwurf needed a way to establish presence without encumbering the local computers or the networks with unnecessary computation or traffic.

2.1 Presence

The Netzentwurf platform was based on ASP web pages attached to a database. This means that the web pages a user sees do not exist as separate files, but are generated "on the fly". This allows the web pages to be dynamically generated according to the actual state of the database information. The use of ASP and the database meant that we could track the individual users, provided we could identify them. Thus, we made the decision to have students, teachers and other users identify themselves by logging onto the platform. At the same time, we redesigned the interface to allow different information to appear in separate areas on the screen.

Figure 2: Netzentwurf Console

Other virtual design studios have incorporated logon mechanisms and user tracking, but in a separate window. (Johnson 2000) The student's responses to the dispersed nature of earlier Netzentwurf projects led to the decision to try and collect the various aspects of the Netzentwurf experience into a single place or window. The metaphor of a console was used in the graphic design.

In the project, we refer to the Netzentwurf as the entire program of virtual design studios. The platform is the system used to support the programme including the web, ASP and database servers. The console refers to the graphic interface used to display the information.

The Netzentwurf console is divided into three areas: a menu, a workspace and an awareness bar. The menu provides a two layer series of user options, which remains stable. In the contexts of Base, Comm (for communications), Tools and Projects, the user is offered items that change the content of the workspace. The workspace itself is an open slate that displays information according to the context. Thus, in the Chat session, the Chat text is displayed here. In the logbook, the calendar appears here. The awareness bar provides a list of the users who are currently online and, like the menu area, is stable and remains continually visible. (See Figure 2.)

2.2 Awareness

By having the users log into the system, the apparent anonymity of the Internet is traded off for functionality. The initial hurdle to logging onto the system was made as simple as possible by requesting only the name and email of the user. In fact, the platform is completely open in that anyone can register himself or herself on the platform. Once the user is registered, he or she can log onto the platform and then are greeted and linked to the console base workspace. From the base, they can navigate to the other areas of the platform in order to join a conversation or view the work of a student group.

In the background, the server initiates a reload of the awareness area of the console every two minutes. Thus, if other users have logged onto the platform, their presence is displayed to all users. Should a user have logged off they will cease to appear in the awareness area. The reload of this area (an HTML frame) also means that the platform can check if an HTTP request is made from the IP of a user within the last two minutes. Should the platform fail to receive an HTTP request, the user is considered logged out. This mitigates problems caused by network failures or when users simply quit their browser without logging off of the platform. Thus, only the users who are truly online on the platform are indicated. As an aside, there are no hidden modes of logging on. System administrators, tutors and students alike are displayed; there is no lurking.

The limited pixel space available in the awareness area led to a weighted display of users. Users are displayed in order of their programmatic proximity. Thus, members of the same design team would appear before members of the same design theme who appear before members of the same University. This allows users to see more or less the people they wanted to see, if they are "there".

3. CREATING A COMMUNITY

A further aspect of the awareness area is that it is interactive. Each user is identified with a mugshot of their choice, their name and IP number. By clicking on their name, a user is shown the personal information about that person in the workspace. The extent of the personal information is, like the mugshot, left to the individual users to fill out, should they wish to. Clicking on the IP number was intended to initiate a NetMeeting videoconferencing session; however, firewall and heterogeneous operating systems have put this functionality on hold until a more robust mechanism is found.

When a user clicks on the mugshot of someone displayed in the awareness area, they indicate their wish to communicate with that person.

This is done through the "talk", which is an HTML driven chatroom. The awareness area of the second person will then display that they have been invited to go to talk. Clicking the Comm button and then the talk menu item brings a user to the talk. (See Figure 3.)

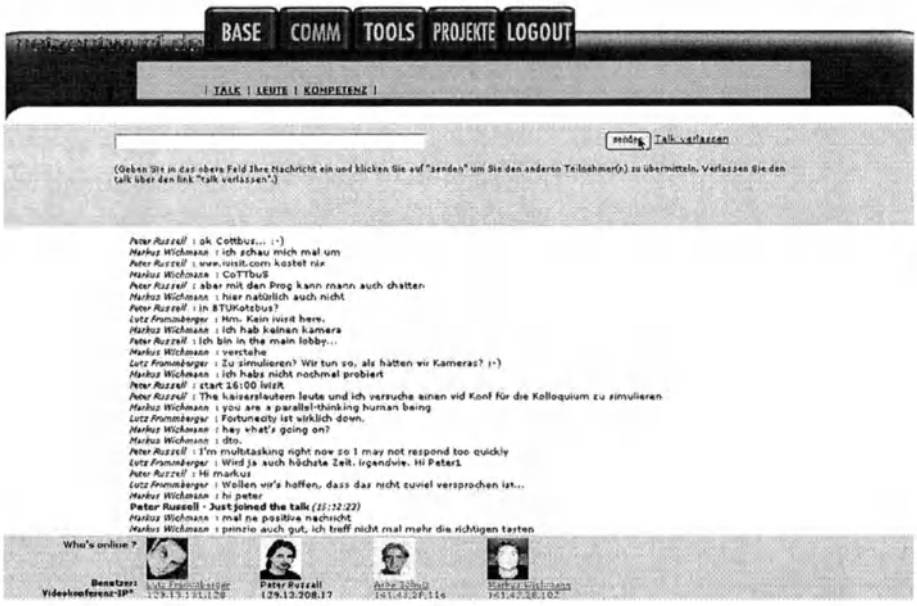


Figure 3: Netzentwurf Console with Talk in progress.

The transparency and immediacy of the awareness area proves the visual equivalence of the voice on the telephone. The mugshot serves as a placeholder for the presence of other users. Their exact appearance or physical whereabouts are unnecessary in establishing their presence. Rather, the indication of their presence combined with the relatively often refreshment of this information enables the users to establish a spatial presence without space. They are there without being anywhere. Additionally, the users are present without being preoccupied, as is the case of chatrooms where one must continually type in order to establish a presence.

3.1 Functionality

The platform serves to act as a place through the transparent display of the users who are currently present. This central feature is augmented by functionality, which helps to bring members to this place. These functions are primarily for the support of the studio work and are oriented towards facilitating communication.

The primary communication augmentation is the "talk" function. This is a modest Chat system that uses the ASP structure of the platform to replicate systems like icq (the Chat protocol named "I Seek You"). The ASP system allows an almost universal web access at the cost of slower response and visual acuity problems with more than 8 members. The chatroom or "talk" functionality is carried out through refresh calls to the browser, which updates the discussion displayed in the workspace. The discussion is, in effect, a series of text string entries in the database. When a user adds to the Chat, their text is entered in the database and displayed at the next refresh.

Other functionality intended to assist the students included a consultation scheduler where the students could make appointments with tutors. These lists are posted by the individual tutors and then students can sign up for a consultation period and receive a "desk crit" from their tutor, either in person or online.

Additionally, the students could use the directory functions to contact other members of the platform or use the mailing lists to send information to groups of students. These directories were often used simply to identify people and help put names to faces. (See Figure 4.) A newsgroup set up to allow for asynchronous communication went mostly unused. A schedule listing lets the tutors post important meetings such as mid term and final reviews.

Figure 4: Netzentwurf Console with Mugshot Gallery

3.2 Limits

The summer semester of 2000 showed that students and other users usually go to the chatroom as a meeting place to co-ordinate further discussions. This is mostly due to the fact that the talk function is too cumbersome to carry out longer discussions. After about 8 users, the refresh rate is too slow and too unsettling for the eyes to allow discussions to flow freely. However, its immediacy coupled with the current information in the awareness area was quite useful in initiating informal or impromptu discussions. These usually took place on an irc or icq channel. Web cams were used by a few students to carry out videoconferences with freeware software from iVisit. As well, the continually lowering costs of telephony allowed the students to simply call one another on their almost ubiquitous cellular telephones.

Mid-term reviews were attempted using iVisit videoconference software and VNC shared screen software, which are both freeware products. A room with two video projectors was established at each institution so that the students could see a shared web browser on one screen and a communal videoconference on the other. The technical problems such as bad bandwidth, firewalls and different operating systems as well as the scheduling hurdles involved in three universities served to torpedo about half of these meetings. The individual communication among the design teams met with less resistance.

What is more telling, is that some of the student design teams decided that they needed a face to face meeting and arranged among themselves to meet (Cottbus and Karlsruhe being 800km apart). This was not prohibited, but it was thought that the distances would be too large and thus would force the student groups to use the Internet as their communication medium. The fact that they underwrote these meetings themselves shows how strong the need for face to face communication is, in spite of the hurdles of time and distance. Essentially, the bandwidth offered by videoconferences and the like is nowhere near the bandwidth of a face to face meeting with a roll of sketch paper.

The students also called for a revised logbook function that has, as of the winter semester 2000-2001, been implemented. The logbook allows students to enter the URL of new pages they have created with a date and description. Thus, users can see what has changed in the last days with individual students or with the entire platform. This is especially useful in the later stages of the semester when it becomes difficult and time consuming to browse through the student work trying to ascertain what, if anything, is new or has been changed.

4. CONCLUSIONS

The goal of establishing place has largely been met. The organisers took a chance in trying to use a low profile way to signal the presence of platform users, but this has surpassed our expectations. The acceptance and intuitiveness of the awareness area has been successful. There were and are students who did not appreciate the potential of the mugshots and ended up displaying pictures of movie or sports stars. However, the questionnaires carried out at the end of the semester showed an overwhelming positive reaction to the console metaphor and the platform in general. Indeed, the acceptance of the platform has led to a long wish list of additional functionality on the part of the students.

4.1 Directions

With the implementation of the logbook and competency functions, the student-oriented side of the platform is robust and relatively complete. What are now needed are tutor-oriented tools that will allow a simplification in the preparation, offering, assignment and evaluation of individual design themes. With such tools in place, the platform will be more or less run by its users, independent of ifib, which is far from the case at present.

Ideally, interested institutions can then register with the system and then offer their studios to whomever they wish. Interested students can then apply to take part and, given reciprocal recognition, participate in the studio. This will allow the implementation of the independent architectural design studio market (Russell and Forgber 2000).

5. REFERENCES

Forgber, U., P. Russell, 1999, "The Virtual Design Studio", *Proceedings of the EAAE Annual Conference, Plymouth, UK,* University of Plymouth, Plymouth

Russell, P., N. Kohler, U. Forgber, V. Koch, J. Rügemer, 1999, "The Virtual Design Studio as an Architectural Graphics Laboratory", *Turing to 2000: The 17th Annual eCAADe Conference, Liverpool, UK*, University of Liverpool, Liverpool

Koch V., P. Russell, 2000, "VuuA.org: The Virtual Upperrhine University of Architecture", *Promise and Reality: 18th Annual eCAADe Conference, Weimar, Germany*, University of Weimar, Weimar

Russell, P., U. Forgber, 2000, "INDeS: Interuniversity Networked Design Studios ", *Promise and Reality: 18th Annual eCAADe Conference, Weimar, Germany*, University of Weimar, Weimar

Elger, D, P. Russell, 2000, "Using the World Wide Web as a Communication and Presentation Forum", *Promise and Reality: 18th Annual eCAADe Conference, Weimar, Germany*, University of Weimar, Weimar

Johnson, B., 2000, "Sustaining Studio Culture: How well do Internet Tools Meet the Needs of Virtual Design Studios?", *Promise and Reality: 18th Annual eCAADe Conference, Weimar, Germany*, University of Weimar, Weimar

The Netzentwurf is located at *http://www.netzentwurf.de*
The platform is open and anyone is welcome to join and partake in the open studios.
A variation of the Netzentwurf is at the Virtual Upperrhine University of Architecture located at *http://www.vuua.org*
iVisit software is available at *http://www.ivisit.com* and is free of charge. It is available for Windows and Macintosh operating systems.
VNC software is available at *http://www.uk.research.att.com/vnc/* and is available for almost all current operating systems.

Capturing Place:
A Comparison of Site Recording Methods

Nancy Yen-wen Cheng
University of Oregon

Key words: Fieldwork in Architectural Practice, Design Process, Computer Media

Abstract: When designers document locations for site-specific projects, how do tools affect recording of visual data? We observed design students visiting future project locations with sketchbooks, cameras and video and analysed the resulting Web-based field reports by tallying images according to scale and content. The study describes how tools shape place-recording phases and explains how field reports can contribute to understanding the tools. Examining reports from different classes exposed the importance of objectives and setting characteristics in shaping data collection. A refined approach for studying new place-recording tools is suggested.

1. INTRODUCTION: TOOLS FOR PERCEIVING PLACES

As experiences of natural and urban environments are displaced by technology-mediated experiences, our need to savour and capture authentic moments increases. After sitting in front of a computer screen all day, even a walk through a parking lot is flooded with stimulating kinaesthesia and evocative sensations. Capturing the sensuous experience of place into a tangible form is a challenge made more enticing by new gadgetry. How can we go to a place and fully convey its essence to someone who is not there?

Since this impulse to document place lies with journalists, geographers, urbanists and artists of all types (Hiss, 1990), it is important to distinguish the needs of the environmental designer. Corner (1992) explains that the challenge of representing environments starts from the size, complexity and richness of the physical world and our limited capacity to record, absorb and

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 243–255.
© 2001 *Kluwer Academic Publishers.*

process the flood of information. We are further challenged by our tendency to reduce the full sensory experience into visual representation.

While each person will approach a place slightly differently depending on what is sought, there are archetypal objectives that shape what will be captured. For example, a tourist looks for beautiful, famous or unusual photo opportunities. An engineer seeks relevant clues about building performance. A real-estate agent searches a property for marketable labels. A designer's interest lies somewhere on the continuum between technical assessment and artistic study, since both quantitative facts and qualitative impressions are sought.

By visiting a site, a designer collects information about physical, social and cultural conditions while perceiving nuances that may shape design direction. Ideally, methods of place recording heighten perceptions and strengthen understanding of a location. But as less efficient processes like sketching give way to a new array of techniques, how can we maintain or enhance the thinking eye? Tools such as video cameras, 3D digitisers, and motion-capture devices automate ways to capture a large amount of information efficiently, but do not guarantee a thoughtful process. Since technology influences how we see, it also shapes what we see and how we think. We need to better understand tool biases so we can target their use in situations where they could increase awareness.

To start understanding the influence of tools on the process of recording environments, this study compares traditional sketching and digital photography. After preliminary observations of the process and how it varies with tool usage, we analysed site information distilled onto Web pages by looking at what information can be gleaned from tallying imagery. Preliminary correlations between tool use and content are described along with limitations of the results, and suggestions for further study.

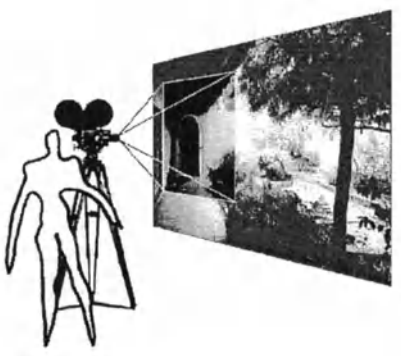


Figure 1. Place recording needs to consider man, media and environment

1.1 Rationale for the study

While we can observe qualitative differences in site recording with different tools, it is difficult to track operations and correlate them to thinking. Rather than examining a few designers' process, this study surveyed a greater number of designers' products. While interviews or talk-aloud procedures could give a better understanding of the connection between media process and thinking, they were likely to reveal individual idiosyncrasies.

Instead, site information as published on the Web was examined. Because of its accessibility, if it proved useful others could easily and efficiently peruse many cases. As material selected for further study, the Web pages have a special significance in the site-recording process. Even if the images were chosen with little intention, they acquire importance as a substitute for the site in the subsequent design process. Like an amulet or religious icon, the images hanging over a workspace or posted on one's homepage gain significance after repeated viewings. (Downing 2000) For these reasons, examining the presentation images for ideas on media bias was worth a try.

2. BACKGROUND:

2.1 Phases of site recording

Gathering and presenting place information takes place in the beginning of the design process continuum according to Crowe and Laseau (1984). It makes up part of the Recording and Analysis phases that precede Design. Site recording usually contains some form of 1) Pre-trip preparation, 2) On-site documentation, 3) Post-trip Analysis, 4) Presentation, 5) Reflective use in Design.

Prior to the trip, a designer needs to plan what information will be gathered, how the site will be toured and how team members will be deployed. Equipment and existing documentation needs to be gathered and reviewed so that precious time at the site is used efficiently.

A designer comes to the site with intentions and expectations that may need to be modified at the site. Unfamiliar terrain makes it impossible to fully predict what will be worth recording, so a designer needs an alert eye to catch the unexpected. (Crowe and Laseau, 1984) Particularly for group efforts, a method for organizing and storing ideas, images and video clips is needed. Information needs to be organised in a retrievable form with enough

identification & cross-referencing to be useful. Data can be arranged by format, narrative sequence or location so that it feeds naturally into a planned presentation. (Ehrhardt and Gross, 2000)

On returning from the site, the information's completeness should be reviewed to determine the need for further site visits. The information may be collated like a jigsaw puzzle, or interpreted into diagrams so that patterns can be seen from the fragments. By analysing highlights and deficiencies, design opportunities can be identified.

The results can be presented simply, as in pinned-up photos, or elaborately, as in interactive multimedia websites. Expandable formats foster a site description that becomes more complete from revisiting a site over time (Lynch, 1972). Web presentations can be adaptable by centralizing information and inviting online contributions.

Throughout the recording, analysis, and presentation stages and then during the design process, the artefacts of site information feed reflection about how to create a responsive design solution.

2.2 Collection phases vary according to media

Observing students on site visits revealed how each site-recording phases is shaped by tools employed. For example, at the site, students sketching had long periods of seated reflection at a few selected places, listening to nature and observing subtle details. In contrast, those with cameras moved freely through the site, covering much more geographic area, gaining a richer haptic experience.

Because methods generate different kinds and amounts of raw data, they require different kinds of post-processing. Slower methods of recording, such as sketching, might lead to more on-site reflection but yield less data at the end of the day. Quicker methods, such as photography and video, may curtail meditative pauses, but record great amounts of data that facilitate reflection afterwards. The sketcher can walk away from the site with finished product while a prolific photographer or video team needs to put time and care into editing. While editing a large number of images or video segments can be time-consuming and cumbersome, the resulting presentation can contain much more information than sketches. Photos and video can show information in vivid detail, providing a comprehensive record for verification and enrichment.

In contrast, serendipitous experiments and idiosyncratic sketches may lack copious amount of objective information but can provide a personal site interpretation that feeds the design process. More intuitive onsite experimentation can be fostered by the expectation of a simple editing process.

2.3 Trade-offs with new tools

Many issues about traditional tools, such as the trade-off between collection time and editing time, extend to new tools. Tools that assist in speedy collection of raw data collection require additional tools to rectify, consolidate and interpret data. 3D scanners such as the environmental Cyrax system quickly read complex forms with precision by measuring the time of flight for a laser pulses. At a smaller scale items (shoes to cars), laser-stripe triangulation scanners, such as the Cyberware equipment used for the Digital Michaelangelo project, generate surface profiles by measuring from an oblique view the distortion of a laser line as it crosses a raised or depressed surface.

The resulting masses of digitised data require filtering into a usable form. Akin to raster to vector conversion, point clouds from 3D scanners must be grouped into polygons for efficient rendering and into geometric forms or NURBS surfaces for controlled modelling. Simpler collection methods can substitute for automatic acquisition. For example, desktop digitisers by Immersion and others allows manual point by point input of 3D coordinates, slowly generating a digital model from physical from. While the method lacks the speed of laser scanners, the sparser data can be input in a logical way, requiring no filtering but some error checking. Collecting sparser but more crucial and more organized data saves editing time afterwards.

Tools to consolidate fragmentary data and confirm consistency can increase efficiency by identifying errors onsite. Tools like PocketCAD for Windows CE and AutoCad View provide simple drawing and mark-up capabilities on palmtop computers so that measurements and annotations of existing conditions can be combined onto one file. Individually collected information can be shared through wireless devices.

On returning from the site, other tools can assist in making the collected information useful. Photomodeler mimics more expensive photogrammetry systems in generating 3D models by having the user pick out features that are common to photos taking at different vantage points. Tools like Erhardt and Gross' Placemaker (2000) help organize place images for the Web, keying annotated photos and panoramas into an orienting key plan. By providing a logical format, the tool assists users in creating a professional multimedia presentation.

As preparation to studying how these new tools affect the site recording and publishing process, the author and assistant, Katalin Czege, compared readily available tools.

3. METHODOLOGY

We prepared for guiding students site visits through preliminary trials with sketchbook, digital camera + audiotape and videotape. The trials provided a basic understanding of logistical constraints, procedural mechanisms and perceptual influences. We inspected a variety of digital place-based presentations and created our own Web field report on a place. We then sent students on site visits with sketchbooks and cameras (and in one case video) to capture place information. The reports contained what the students chose as the most relevant, characteristic, legible or memorable information and summaries of what they thought about the site. We inspected these reports for the influence of the tools.

Would the differences between media would be evident in the Web pages created from the site visits? We conjectured that counting the kinds of images in the presentations could give a quantitative look at media's influence.

We sorted the media broadly with the Sketch category including watercolour, charcoal, pencil and pen. While we could guess what was drawn on site, it was not clear which sketches were made from photographs. Likewise, since we couldn't clearly distinguish between scanned film and digital camera images, they were both considered to be in the Photo category. Video was taken with an analogue video-tape-recorder. While it was would have been desirable to capture audio notes, sound was only captured on the videotape.

To parse the captured information, we had to invent categories. Initial thoughts to sort images by content (spatial order, human activity, natural forces and cultural meaning) proved too subjective. The image counts mirrored the site locations closely rather than revealing about media types. Instead, we chose to distinguish architectural versus natural subject and estimate the scale of the image. For scale, the imagery was sorted according to the distance of the viewpoint to target of interest. The analysis spreadsheet contained the following categories:

Table 1. Categories for logging websites

Identifier	Name	Login name identifying website
	Group	Course number and instructor
	Medium	Tool used for recording information
Scale and subject	Site Vistas	Long urban views and panoramic images
	Site Elements	Middle-distance images of natural components
	Site Textures	Close-up shots of natural elements
	Architectural Forms	Complete buildings & overall views of man-made forms
	Architectural Elements	Middle-distance images of man-made components
	Architectural Materials	Close-up shots of man-made objects

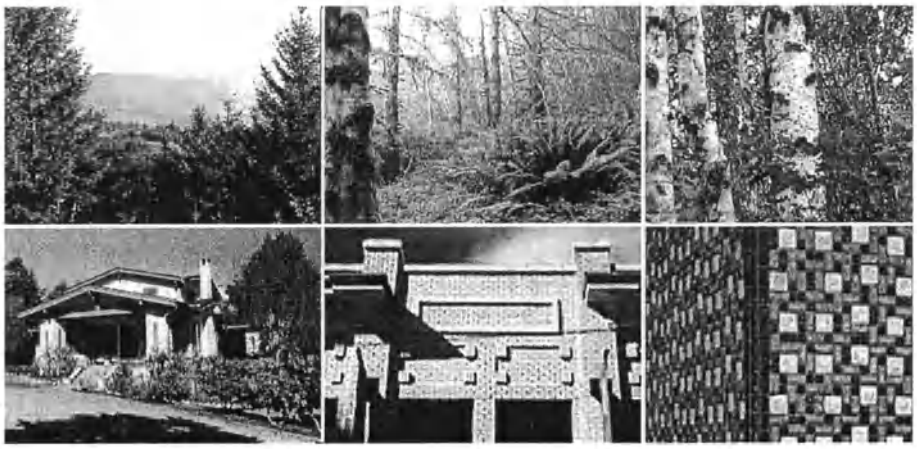


Figure 2. Categories shown in photos: Top: Site vistas, elements & textures, Bottom: Architectural forms, elements & materials

3.1 Context: designers, duration, site type, objectives

In each case, architecture students in a first professional degree program with basic Web authoring training, collected information at their studio project sites for a few hours and then summarised the information over a week or two.

In the first group, 30 students visited a natural undeveloped hillside to find and record the site for their upcoming studio project. In the second group, 21 first year graduate students in a computer graphics class visited their studio sites, individually or in small groups. Only those in the same design studio designing for an empty lot were included in the study. For comparison, we also looked at a third group, 85 second year undergraduates, who had gone together to an urban site with specific issues to address and a fourth non-digital group who created printed rather than online reports.

The first group generated 10 pages with photos (average 2.9 photos) and 8 pages with sketches (average 5.0 sketches) and one edited video. The second group created 21 pages with photos (average 2.4 photos), 2 pages with sketches and photos (average 3.5 images). The third studio group, working in groups of about seven students, created 18 more elaborate reports (average 7.7 images). The fourth group created 5 pages with sketches (average 2.6 images) and 6 pages with photos (average 7.0 images).

4. DATA: WHAT DID THE TOOLS CAPTURE?

In the first group, compared to the sketchers, the photographers concentrated on more natural elements (86% to 68%). This could have been due to the fact that groups gathered and rested close to built structures, allowing time for sketching. For this case, both groups concentrated on either the very large scale or on very small scale (primarily natural textures). At the middle scale, both groups registered few examples of natural elements (one case or 3%) compared to architectural elements (8%). The designers saw natural elements as a part of a larger whole, whereas perhaps due to their training, they recognised architectural elements as having a more pronounced character worthy of highlighting.

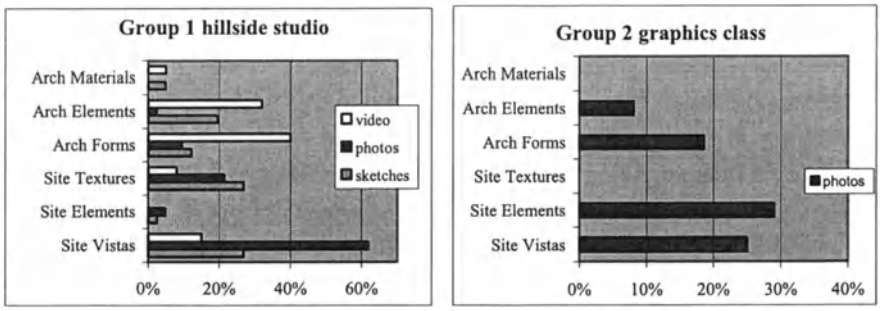


Figure 3. Image tally by media for groups 1 & 2

Students in the second group opted to use digital cameras over scanning sketches perhaps because it was faster and the class had a digital agenda. They concentrated on site elements (29%), and vistas (25%) with less attention to complete building forms (19%) architectural elements (8%).

Because of the third group's agenda to look at urban continuity, they recorded large-scale information (80% of sketches and 82% of photos) much more frequently than medium or small-scale information. In comparing use of photos vs. use of sketches, students used photos much more than sketches for the large-scale site vistas, especially when they contained natural elements. Sketches, by contrast, were used for building scale pieces, with some drawn from photographs. Students found it easier to draw the regular geometric order of man-made forms rather than the complex chaos found in nature.

The fourth group went to a site that was primarily natural with adjacent buildings primarily on one side. This was reflected in the dominance of the site images (88% of the photos & 85% of the sketches) over architectural images. As with the other groups, photographs were used more than

drawings for site vistas and drawings were used more for identifiable objects (site elements and architectural elements).

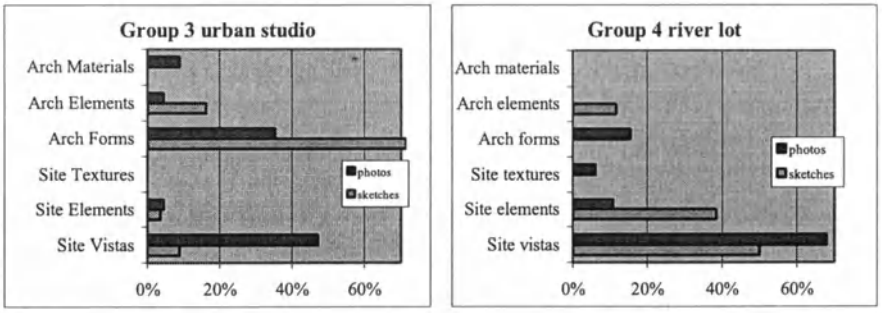


Figure 4. Image tally by media for groups 3 & 4

4.1 Qualitative aspects of the Group 1 video

Video from the first studio group visit was consolidated into a single tape. So while it is statistically insignificant, the video was logged to try a comparison method. Segments of the video were labelled with one of the six scale and content categories, and according to the duration of each segment, a percentage was assigned to each category. This procedure made it possible to compare video to the photos. The video's anomalous emphasis on architecture over nature (72% vs. 18%) reflected that its ability to work better than film cameras under low-light interior conditions.

Reviewing the video produced the following qualitative observations. It gave a very vivid sense of capturing an ephemeral moment because of it arbitrarily captured people in specific activities with bits of particular conversations. The imperfect shaky camera and occasional voice-overs gave a stronger presence to the author than still images. Spatial adjacencies and rough orientation came through, but absolute relationships were difficult to perceive.

5. DISCUSSION: WHAT DID WE LEARN?

In looking at all the groups together, there is not a clear correlation between the recording medium and tallied report image categories. Within the wide variation of media use between groups, we observed a few tendencies. The students used photography for subjects too complex to draw, such as urban panoramas and organic textures. In all media, they

highlighted things they knew well, such as architectural elements, and let less interesting pieces fall into larger views. They displayed both assigned information, such as building facades, and their own choice of engaging details. To generalize, people capture a subset of what is at a location depending on what they are looking for and their ability to find and recognize it. Individuals will do different things with tools depending on their training, talent, and interests.

The study revealed the role of influencing factors in place recording and the study of place recording. Among the many variables in the site recording process, the subject's intentions and the character of the site appeared to be the most critical factors in defining what is collected. The type of tools and students training followed in importance. So to understand media variation, it is crucial to hold send all subjects to the same sites with the same directions.

5.1 Media Constraints & new tools

Observing and trying place-recording methods accentuated how each medium engages the user to tailor work to its nature. The tools invite us to make an appealing artefact and control how this can be done. "Every type of visual, numerical, and verbal representation follows its own logic, "talking back" to the designer and clouding the relationship between representation and reality." (Bosselman 2000) Creating a pleasing composition becomes as important as recording important information. Circumstantial details like fleeting sunshine can make secondary forms inviting. Conversely, some subjects do not fit some techniques. Silence on audiotape or stillness in video compels us to create drama or motion, vast repetitive fields challenge sketch artists.

While each tool frames its results, simpler tools tend to be more versatile and high-tech tools more constrained. Sketchbooks can carry representational images, analytic diagrams, and text in idiosyncratic ways, but their digital equivalents such as personal digital assistants, constrain input techniques more narrowly. As new tools are precisely tuned to specific tasks, tool selection becomes more critical. Just as tight interiors require a wide-angle lens, situations can demand specific kinds of tools.

The specialization of each medium means that resulting products cannot be parsed in the same way for analysis. The nature of a tool's raw data and its manipulation must be considered in guiding the site recording process and in characterizing the resulting products. Perspective images are more naturally categorized according to pictorial aspects, such as viewpoint distance, than by logocentric content categories. Sentences can be sorted into abstract content categories (spatial, cultural, natural, cultural) more

easily than images since text articulates conceptual thinking more clearly than graphics. Examining additional place videotapes (Kellett & Girling 2000) confirmed that a measurement's usefulness is dependent on the medium. While it was possible to translate the image tally to video time segments, tallying viewpoint distance became less interesting after the cameraperson standardised shots to fixed-location zooming and panning to reduce camera shakiness.

6. FUTURE WORK: METHODICAL AND ARTISTIC APPROACHES

In this round, tallying web page imagery was more useful for revealing a group's site recording interests than for showing tool bias. With modification, field report analysis could be more informative about the media's influence on vision and perception. Comparing concrete factors such as the perceived dimension of represented vs. real objects could be more fruitful than the image tallies. Supplementing website analysis with interviews or thinking-aloud sessions would illuminate more of the process-process connections. (see Herbert 1993 & Robbins 1994) For new tools, the protocol could include using subjects to review the created material:

1. *Preparation:* Make pilot trials with audio taped notes, train students in using tools.
2. *Field Visit:* Design students visit a compelling place with different toolkits using audiotape annotation, then summarise findings for the Web
3. *Survey:* Web authors are queried about site features to track site perceptions and memories.
4. *Review:* Other students examine the Web reports; describe differences in how the presentations capture sense of place and scale, before and after visiting the site.
5. *Analysis:* Web pages, surveys, audiotapes and student reviews are examined for robustness of place description and accuracy of scale depiction.

With the long-range goal of defining task-appropriate toolkits, this study begins to document how tools affect field recording and examines one way to look at Web-based site documentation. Related investigations include:

- Refining the methodology for studying site-recording,
- Comparative testing of recording tools
- Examining media's role in successful site-specific designs
- Tracking representations in site perceptions during the design process
- Developing more robust representations.

Figure 5. Media shapes what is captured: sketch shows abstract concept of alders

Both methodical research and creative exploration can contribute to our understanding of place representations. In a recent study on the representation of non-visual site information, Robitaille (2000) explored collage techniques to record sensations of touch, sounds and smells in an environment. This type of artistic approach can be appropriate because a designer needs not only factual information, but also details shaping the gestalt of a place. Randolph Hester, a landscape architect, described on-site sketching as "visual listening: looking so carefully that you pick up essential spatial details that create the uniqueness of a site.... Active meditation reveals the essence of a place, the soul that touches your heart the way that office drawing cannot." (1993) The ephemeral moment of long diagonal shadows or a squirrel jumping across a frame can strike a chord and bring the designer back to that moment of being there. Part of the job of understanding a site is becoming aware and open enough to see the unexpected, to relish the moment of just experiencing what happens.

So a balance needs to be struck between rational procedures and intuitive gathering. Checklists of site information topics (White 1983) can make examinations more comprehensive, but may constrain observations to those expected from traditional tools. Too tight a recording protocol would make it difficult to pick up serendipitous events that stimulate design. Rather than defining what should be found, we should concentrate on defining procedures for searching. In this way, we can guide site surveys to be comprehensive and efficient while fostering the circumstantial perceptions that can spur design thinking. Reviewing the student websites showed that we see what we look for and we see what we can name. Our challenge is to open our eyes to what we're not looking for.

7. ACKNOWLEDGEMENTS

Katalin Czege, Ronald Kellett, G.Z. Brown and Stephen Lamb helped shape and encourage the ideas in this paper. The University of Oregon Foundation and the Intel Corporation provided support for this enquiry.

8. REFERENCES

Bosselman, Peter, 1998, *Representation of Places*, Univ of California Press, Berkeley, p. 187

Corner, J., 1992, "Representation and landscape: drawing and making in the landscape medium", in *Word and Image*, 8(3), p. 243-275.

Crowe, N. and P. Laseau, 1984, *Visual Notes,* Wiley & Sons, New York

Debevec, Paul, http://graphics3.isi.edu/~debevec/

Digital Michaelangelo, http://graphics.stanford.edu/projects/mich/

Downing, Frances, 2000, *Remembrance and The Design of Place*, Texas A&M Press, College Station, TX.

Ehrhardt, M. and M. Gross, 2000, "Place Based Web Resources for Historic Buildings", *ECAADE 2000*, pp. 177-179

Herbert, D., 1993, *Architectural Study Drawings*, VNR, New York.

Hester, R., 1993, in "Portfolios [drawings of landscape architects]", in *Landscape Architecture*, May 1993, 83(5), p. 57-63.

Hiss, T., 1990, *The Experience of Place*, Knopf, New York

Kellett, R. and Girling, C, 2000, *Elements of Neighborhood CD-ROM*, Eugene, OR

Lynch, K., 1972, *What Time is this Place?,* MIT Press, Cambridge

Robbins, E., *Why Architects Draw,* MIT Press, Cambridge, MA.

Robitaille, Sophie, 2000, *Make Sense*, unpublished M.L.A. thesis, University of Oregon

White, E. T., 1983, *Site Analysis*, Architectural Media, Ltd., Tallahassee, FL.

Space Pen

Annotation and sketching on 3D models on the Internet

Thomas Jung, Mark D. Gross, Ellen Yi-Luen Do
Design Machine Group, Department of Architecture, University of Washington, Seattle, WA 98195-5720, USA

Key words: Collaboration, Annotation, Java3D, VRML, and Sketch.

Abstract: Designing a building or a new urban space is collaborative work that involves several people with different backgrounds. To achieve consensus all participants in the process meet to discuss documents such as floor plans and sections. This paper reports on the progress of Space Pen, a new system to allow several users to draw on and annotate a three-dimensional representation of a building remotely over the Internet.

1. INTRODUCTION AND MOTIVATION

1.1 A 3D collaborative environment

Communication between firms and their clients and contractors is becoming increasingly challenging as many projects now involve people and teams from different countries or even continents. The only way for everyone to participate in the design throughout the whole process is to schedule meetings, which can be difficult to set up and therefore months apart. For a client it can be frustrating not to be constantly part of the progression of the design. Yet it is often time consuming for an architect to reconsider and amend decisions that the client might not agree on, or that are too difficult or expensive for the contractors to realize.

One key to improving online communication is to make the project as clear as possible to all the participants. Floor plans and sections are useful to communicate ideas among building professionals, but people without architectural training have difficulty envisioning space and volumes from 2D

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 257–270.
© 2001 *Kluwer Academic Publishers.*

documents. Where before scale models were often used to communicate design ideas with clients, now most architectural firms have the capacity to build 3D models and present computer rendered images to their clients.

When Disney created California Adventure Park, designers created a full 3D virtual model of the design. Disney's officials examined the model, which revealed the building's structure and the different phases of construction in a virtual environment. According to Disney, problems and design issues, totally unobvious from the physical model and 2D documents, were discovered after seconds of browsing around the virtual model (New York Times, 2001).

In this paper we present Space Pen, a pen-based reviewing system that takes advantage of the Web. Space Pen enables clients, contractors or associates to view a 3D representation of the building (or urban space) and annotate those models by writing and drawing on them anytime, anywhere, on their ordinary Internet browser throughout the entire design process.

1.2 A pen-based interface to sketch in 3D

Online critique and collaboration involves people with different backgrounds and different levels of confidence in using computers. A typical architectural firm usually deals with several distinct clients and contractors each year, all with various computer skills and knowledge. Drawing and sketching communicate ideas quickly, clearly and easily. A pen-based interface to a CAD program eliminates the need for great knowledge of structured menus or command line interface, and for precision in the execution. As Garner mentioned in DCNet'00 (Garner, 2000), "the widespread use of sketching in design activity would seem to suggest that it offered appropriate support to both the defining and resolving of design problems".

Space Pen uses a (digital) pen as a single input modality for annotations and drawings. Because pen and paper is the most traditional and intuitive way to draw and review documents, we can expect every user of our system to understand immediately how to leave comments and annotations.

In Space Pen sketches made on the pen tablet are translated into 3D by leaving digital ink onto the nearest surface of the model. The marks remain on the surface and can be seen from any viewpoint as a visitor walks virtually around the annotated surface.

1.3 Related Work

Online collaboration and annotation on 3D documents is a wide area of research that has application in various fields, especially in architecture. We briefly describe below related work in the use of 3D VRML (Virtual Reality Modeling Language) for design collaborations on-line annotation systems, and 3D input interfaces developed by both research laboratories and commercial software firms.

Christopher Peri at UC Berkeley taught students collaborative design by interacting in a VRML world instead of chipboard models (Peri, 2000). His experiment demonstrates that VRML models can replace physical models to support a discussion of design issues in a project.

Web PHIDIAS (McCall, Holmes, et al., 1998) enabled users to annotate 3D VRML models on the Web using the PHIDIAS hypermedia system to organize the comments. Craig and Zimring's system (Craig, Zimring, 1999) annotated a 3D virtual courthouse with symbols such as arrows or circles. Like Space Pen these systems use 3D VRML models to support collaboration over the Internet. However, none of these systems can import VRML models directly from standard 3D-modelers, and in order to use them, an architecture firm would require extra personnel to perform VRML translations and encoding.

The ProjectPoint (ProjectPoint) and the WebOrganic (WebOrganic) web sites provide services to leave comments and annotations on 2D documents and on Web pages using a regular Internet browser. WebOrganic is more generic; it can be used to comment on any HTML page. ProjectPoint is more architecturally oriented and lets people comment and annotate 2D documents such as floor plans and sections using a browser. The ProjectPoint web site assists the architectural firm throughout the design and construction process by providing allocated server space to share, comment or annotate documents. ProjectPoint is 2D, but it can use Autodesk's VoloView Express to share 3D models.

Autodesk's VoloView (VoloView) and Sigma's eZ (eZ) let you annotate 2D or 3D objects online. However they have little or no walkthrough capabilities and annotations in 3D are actually made in 2D (on a transparent plane in front of the viewer). Both applications set up their own complex environment to provide sophisticated annotation support. They are powerful if the users spend time to learn how to manipulate the models and add annotations. However, we believe drawing-based input will be more intuitive and easier to use for people who don't have time or desire to learn another complex computer application.

Various research projects have investigated sketching as an interface to modify and create 3D models. SKETCH (Zeleznik, Herndon, Hughes, 1996)

is a 3D modeler that uses strokes made with a 3-button mouse to create geometry in the 3D space. SKETCH only understands a finite number of strokes, which makes its drawing environment less intuitive than sketching with a pen on paper. Teddy (Igarashi. Matsuoka, Tanaka, 1999) creates freeform models and rounded objects with a simple and straightforward pen-based interface. For example, you draw an oval to create 3D bubble that users can view and rotate, and/or sketch on top to add more objects, but it cannot create the rectilinear objects that are more often needed in architectural practice. STILTON (Turner, Chapman, Penn, 1999) on the other hand is mainly designed to sketch and create 3D objects made of straight lines over an existing VRML model. With STILTON, unlike Space Pen, the user doesn't draw directly into the 3D space, but on a transparent surface in front of the viewer. The 2D drawing can then be converted into a 3D object and is automatically placed in the 3D world.

These are powerful systems to create 3D geometry from sketches, similar to what designers would do on paper. However none of them combine an easy-to-use pen-based interface in a shared collaborative environment. This is the goal of Space Pen. Space Pen also can import any kind of VRML model without further manipulation, and because it's been programmed in Java3D it will work on any Java3D enabled browser without additional software or hardware.

1.4 Our previous work

Space Pen extends a project we have been working on in the past two years. That project, called Immersive Redliner (Jung, Do, Gross, 1999), is an annotation system to support collaborative review of 3D models on the web. Each user has the possibility to leave a PostIt™ style note (indicated in 3D by a colored sphere) directly on the model. Each sphere is associated with a comment left by the user and both are saved on a remote server. We tested that system on a real architectural project (Jung, Do, 2000) and learned that leaving text comments was not always sufficient to clearly express a reviewer's idea. Often, a sketch or a drawing on the object in question would have helped to clarify and augment the text annotations.

Redliner was implemented in VRML and EAI (External Authoring Interface) and this implementation architecture sharply limited further development. We decided to port the whole system to a much more powerful, new software platform (Java 3D) that would especially support three-dimensional drawing as input.

1.5 Scenario

Space Pen works directly on any regular web browser with Java 1.3 capabilities. When it is completed, architects, clients and/or contractors will be able to use it to review and comment on a 3D representation of the architect's design proposal. Participants log on to the Space Pen web site to view the project in 3D in their own web browsers. They move around the building as they would once it has been built and leave drawing marks and annotations directly onto the 3D model, and save those as part of the model[1]. Each user chooses a different pen color to represent their comments or drawing. Figure 1 shows the Space Pen applet window. Once logged on, the user becomes immersed in the virtual 3D model. On the left is a set of different pen colors for drawing annotations on the model. The user can navigate around the model using the arrow keys of the keyboard.

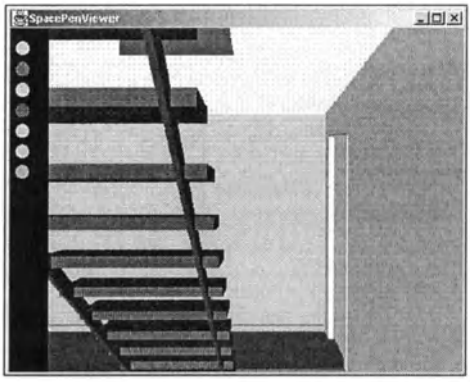


Figure 1. The Space Pen window. The user logs in and is transported inside the model.

Figure 2 shows the annotations drawn by the first user who suggested separating the window in the model into two separate panes. Because this proposed change doesn't need much explanation, the text can be written graffiti-style, directly on the problem area.

[1] At the time we are writing this paper, the saving part of the prototype has not been implemented yet, but using the same technology we used for Redliner, we believe that it can be completed easily.

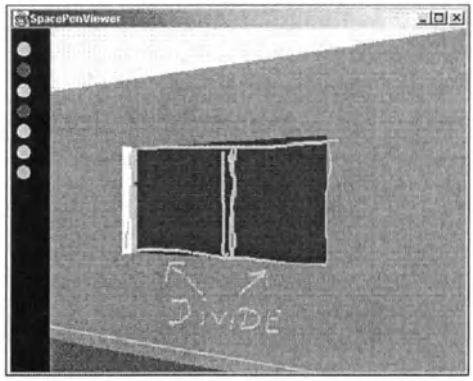


Figure 2. Drawing marks left by the first user.

A second user logs in (Figure 3) and suggests creating two windows instead of one by moving the right windowpane towards the wall. The drawing marks left by each user are embedded directly in the 3D model with different colors and can be seen from different points of view.

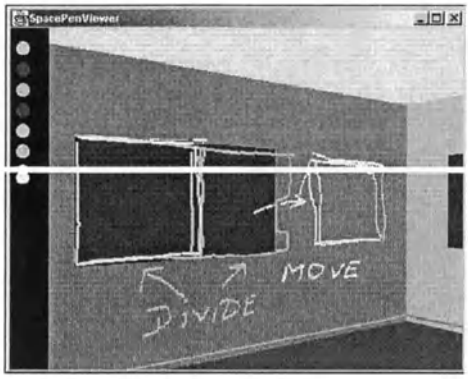


Figure 3. Drawing marks left by a second user on top on the first ones.

Figure 4 shows the comments made by a third user who decided to leave drawing marks as well as a typed-in note associated with a more elaborate text. The line-width of the marks in the 3D model is independent of how far you're standing from them (Figure 4, right). Even from a distant point of view, the marks appear clearly on the building.

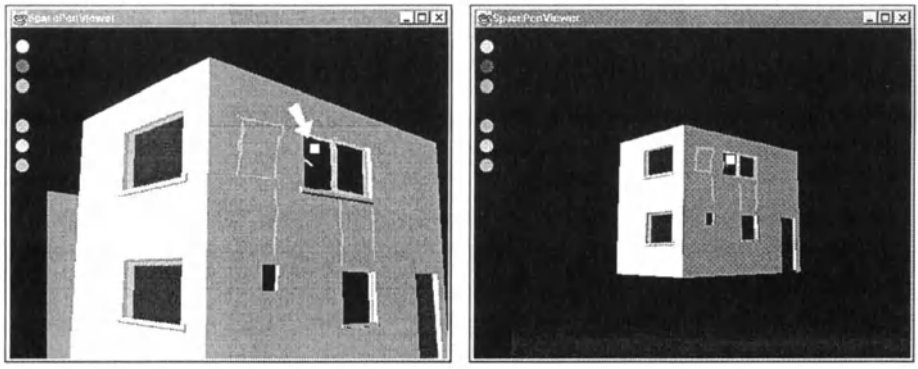


Figure 4. More elaborate comments can be left and will be represented as a PostIt™ note stuck on the model (left, pointed by the arrow), and can be seen even from further away (right).

This simple scenario demonstrates how easy and straightforward the Space Pen interface is to use and how in a few minutes design suggestions can be recorded graphically and in three dimensions. By walking virtually around the building, every user gains an immediate understanding of the project and of the changes proposed by others. Using Space Pen, design issues can be discussed and proposals can be made directly into the 3D environment even if all parties are miles apart from each other.

2. IMPLEMENTATION

Space Pen is a Java 3D applet. It is implemented using the Java 3D API, (Application Program Interface) a set of Java classes that has powerful capabilities for displaying and manipulating 3D objects. In the following we describe some basic features and concepts we used to implement Space Pen.

2.1 Basic Java 3D concepts

Although powerful, with a wide range of possible manipulation and interaction with 3D objects, Java 3D has a complicated way to represent fairly simple concepts. The Java3D API has no pre-defined default values for their 3D objects. Every object must be specified explicitly as clickable, collidable, normalized and so on.

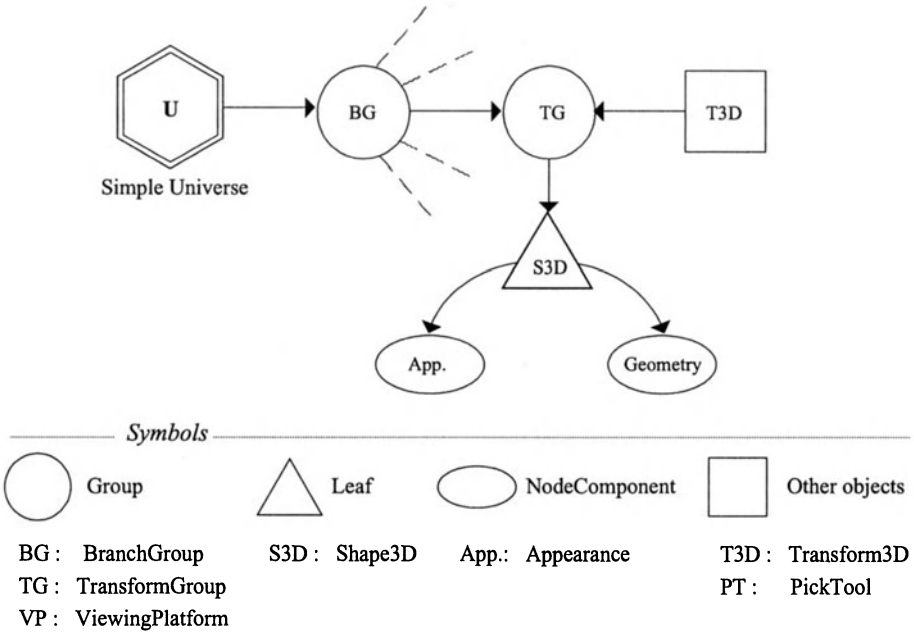


Figure 5. A diagram of a standard Java 3D scene explaining the relationship between each object. On the bottom is a list of symbols and abbreviations used in our diagrams.

Figure 5 shows a diagram of a basic scene implemented in Java 3D. In most Java3D worlds, the top-level object is a Simple Universe (a subclass of VirtualUniverse that sets up the basic environment for Java 3D scenes). The 3D objects are then described in a scene graph composed by BranchGroup (BG) and/or TransformGroup (TG) nodes. Each object has a specific shape (S3D) composed by a geometry and an appearance. More functionality can be added at all levels of the scene graph by adding several other objects. For example a Transform3D (T3D) object can apply transformation matrices on a TransformGroup node. We also used objects like Behaviors or PickTools in the Space Pen implementation.

2.2 The Space Pen implementation

The implementation of Space Pen can be divided into 3 parts:

- Importing and interpreting 3D VRML models
- Drawing and interpreting mouse events
- Walkthrough capabilities

2.2.1 Importing and interpreting VRML models

Although it is widely used, VRML has not become the established standard for 3D on the Internet. However, most 3D modelers can export to this format. Therefore, Space Pen imports and uses VRML models instead of "pure" Java3D models. A VRML Loader Package, part of the Java3D API, imports a VRML model into the Java3D universe.

Nevertheless, we found that a simple import of VRML geometry was not sufficient. We needed to be able to click on an object and have the system to recognize what kind of geometry was selected, the 3D coordinates of the point clicked and eventually the normal of each designated surface. This is not done automatically in Java 3D.

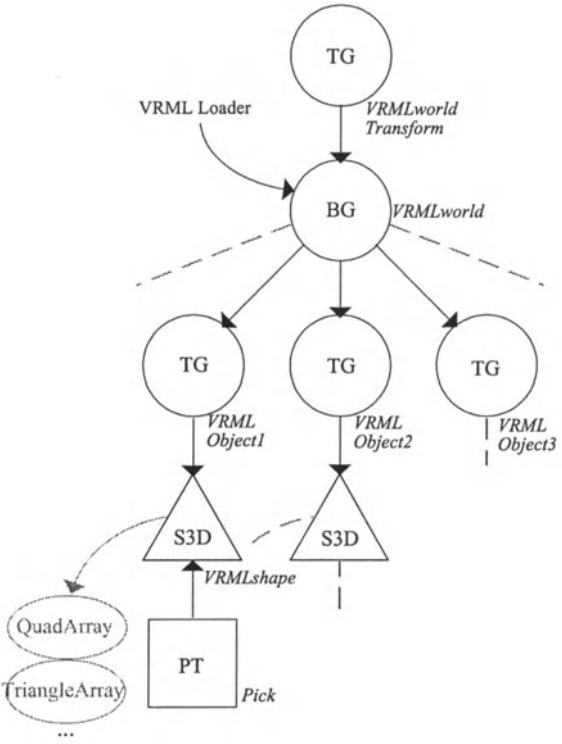


Figure 6. A system diagram showing the importing of VRML objects, the interpretation of each geometry shape and the assignment of Pick interaction.

Figure 6 describes the process of loading the VRML scene, recognizing its shapes and making all the objects clickable (or pickable). In order to be picked (selected), each VRML Shape must be associated with a PickTool object. We need first to count the objects in the VRML world. Then we can loop to analyze each object's geometry. With each geometry (spheres, boxes,

cylinders, planes, triangles…) we associate a PickTool object. Currently, the system only recognizes IndexedFaceSet VRML geometry, which most 3D software uses to describe the entire VRML scene.

Immediately after loading the VRML model, Space Pen assigns every surface in the model full intersection capabilities through the PickTool object, allowing users to select an object and enabling the system to detect which object(s) the user is drawing on.

2.2.2 Drawing on 3D models

Space Pen enables users to draw directly on the surface of the model using a mouse or better, a pen. The system associates two different behaviors with two different pen (or in the Java3D terminology, "mouse") events. Dragging, or drawing on the surface of the tablet, draws lines onto the model; a click event initiates a text annotation via the keyboard.

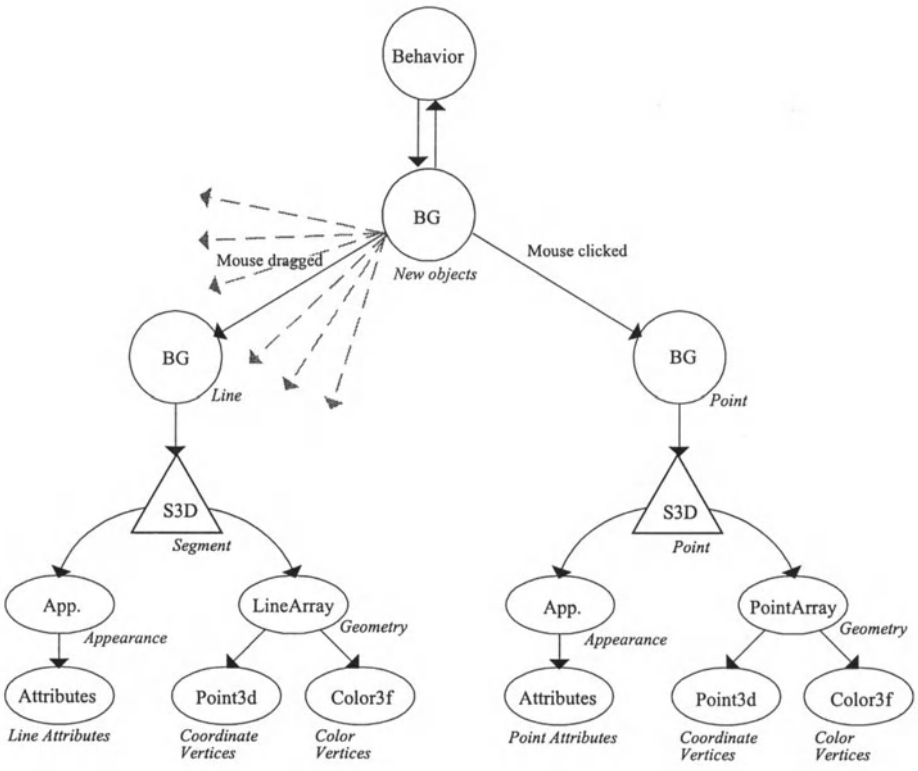


Figure 7. Implementation of the drawing behavior.

The drag method finds the closest intersecting points along the path on a surface. The 3D coordinates of these points are saved in an array and interpreted as key points of a drawing line. The line drawn is a Java 3D object composed of a series of segments defined by 3D coordinate points, color, and boldness attributes.

The click event triggers the leaveComment method that attaches a Point3d object at the click location and brings up a text-input interface for the user to type a more elaborate comment with a keyboard.

Java 3D lines and points have a boldness attribute that sets their width in pixels, independently from how far the user is standing from these objects. This makes comments and drawn annotations visible from most locations in the 3D world.

Figure 7 shows the relationship between all the new objects added into the scene. The behavior object creates a new BranchGroup object that will contain the new lines and comment points. A Segment Shape3D object is created and attached to the Line BranchGroup for each pair of points along the drag path.

2.2.3 Walking inside the model

Java 3D standard methods have pre-set viewing options. The KeyNavigator class and the MouseRotate, MouseTranslate and MouseZoom classes let the user interact with the object in the 3D virtual world. The Mouse behavior classes are used to inspect the 3D model. You can rotate it, move it around or zoom on it using the 3 buttons of the mouse. The KeyNavigator class is a very basic walkthrough Behavior set more adapted for flight simulation than for building investigation (for example, the up and down keys are inverted). None of these were exactly suited to our purpose.

For Space Pen we needed the user to be able to walk through a building or a space, look around and maybe fly. These capabilities were only poorly provided by the existing classes. Therefore we wrote a new VRML browser in Java 3D. The first version of that browser simulates walking and flying behavior controlled by the arrow keys of the keyboard.

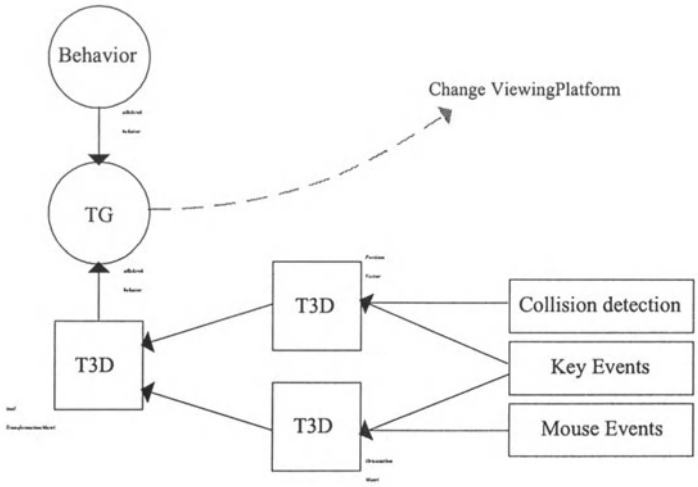


Figure 8. Diagram of the implementation of the walk behaviour class.

Figure 8 explains how we implemented the walk behaviour. A TransformGroup object is created. It takes the result of two transformation matrices as the input, and replaces the ViewingPlatform (an object describing attributes of the main scene graph's view) transformation in the main class. (This class is a derived version of the FlightBehavior class written by Paul Byrne at Sun Microsystems).

Unlike existing VRML browsers, collision detection and gravity are not activated by default in Java 3D. Without collision or gravity, users would start flying when trying to go down and passing through stairs when trying to go up. We resolved that problem by detecting the height of the object immediately in front of the user. If the height of that object is greater than a certain value, then the object is treated as an obstacle and the translation of the user's viewpoint is halted. Otherwise, the object is treated as a step and the viewpoint is translated in the Y direction (up) by a value corresponding to the height of the object.

3. FUTURE WORK

Space Pen is still at an early stage of development. It can load any kind of 3D VRML model in a web browser and visitors can draw or leave notes on these models. We're working on saving annotated models and the integration of all the components: Building on Redliner, we're ready to embed comments, login information and 3D world into a single applet.

However, our goal is more than just reproducing our work with Redliner with sketch input augmentation. We envision Space Pen as a way to manipulate and transform 3D models on the web, using a pen based interface.

Informal observations of people using our system caused us to consider how to make the drawing environment more intelligent. When drawing on non-coplanar surfaces, what we draw is not always what we intended to represent. Our next step will be to process the annotation mark, so the system would recognize which surface the user wants to draw on and what kind of object has been drawn.

We're planning to adapt the 2D drawing recognition technology used in one of our previous projects (Electronic Cocktail Napkin, Gross, Do, 1996) into our 3D Space Pen environment. With symbol and configuration recognition abilities, we could recognize the drawing marks on the 3D as an object or a gesture command for editing and creating 3D geometry. For example, a rectangle drawn on a wall might generate an opening. A rectangle drawn on a horizontal surface could generate a hole or the base of a new column, depending on its size. Symbols such as arrows or textual annotations such as "yes" or "no" could also be recognized and associated with an action to confirm the editing operations. For example, an arrow in an opening might indicate a required dimension; on a door it might assert a move operation with indicated direction and distance.

4. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant No. IIS-00-96138. The views contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

5. REFERENCES

Bimber O., 2000, "A multi-layered architecture for sketch-based interaction within virtual environment", in: *Computer & Graphics, ed. Pergamon*, Volume 24, Number 6, December 2000, p. 851-867.

Brown K., Petersen D., 1999, "Ready-to-Run Java 3D", *Wiley Computer Publishing.*

Craig, D. L. and Zimring, C., 1999, "Practical Support for Collaborative Design Involving Divided Interests", *Media and Design Process, Proceedings of ACADIA '99*, Salt Lake City, p.126-137

Davidson, J. N. and Campbell, D. A., 1996, "Collaborative Design in Virtual Space - GreenSpace II: A Shared Environment for Architectural Design Review", *Design*

Computation: Collaboration, Reasoning, Pedagogy, Proceedings of ACADIA, Tucson, Arizona, USA, p. 165-179

Dorta, T. and LaLande, P., 1998, "The Impact of Virtual Reality on the Design Process", *Digital Design Studios: Do Computers Make a Difference? ACADIA Conference Proceedings, Québec City, Canada*, p. 138-163.

EZ, http://www.ezmeeting.com/

Garner S., 2000, "Is Sketching Relevant in Virtual design Studios?", *DCNet'00 Proceedings, International Journal of Design Computing,* http://www.arch.usyd.EDU.AU/kcdc/journal/vol3/dcnet/garner/

Gross M. D. , 1996, "The Electronic Cocktail Napkin - working with diagrams." *Design Studies 17(1)*, p. 53-70.Gross, M. D. and E. Y.-L. Do, 1996. Ambiguous Intentions. Proceedings, ACM Symposium on User Interface Software and Technology (UIST '96). Seattle, WA: 183-192.

Gross, M. D. and E. Y.-L. Do, 1996. Demonstrating the Electronic Cocktail Napkin: a paper-like interface for early design. CHI 96, Conference on Human Factors in Computing Systems. Vancouver, British Columbia, Canada, ACM. Conference Companion: 5-6.

Igarashi T., Matsuoka S., Tanaka H., 1999, "Teddy: a sketching interface for 3D freeform design", *Computer Graphics, Proceedings, Annual Conference Series, ACM SIGGRAPH'99*, p.409-416.

Java 3D, http://java.sun.com/products/java-media/3D/index.html

Jung, T., Do, E.Y. and Gross, M.D., 1999, "Immersive Redlining and Annotation of 3D Design Models on the Web", *Proceedings of the Eighth International Conference on Computer Aided Architectural Design Futures, Atlanta, USA*, p. 81-98.

Jung, T. and Do, E.Y, 2000, "Immersive Redliner: Collaborative Design in Cyberspace", *ACADIA'00 Proceedings, Washington D.C., USA.*

McCall, R., 1998, "World Wide Presentation and Critique of Design Proposals with the Web-PHIDIAS System", *Digital Design Studios: Do Computers Make a Difference? ACADIA Conference Proceedings, Québec City, Canada*, p. 254-265

Peri, C., 2000, "ARCHVILLE: A Pedagogy for Teaching Collaboration in a VR Environment", *Proceedings of the Collaborative Virtual Environments, San Francisco, USA*, p. 211-212.

ProjectPoint, http://www.buzzsaw.com/content/services/demo.asp

Sun Microsystems Java 3D Engineering Team, 2000, "Java 3D API Tutorial", *Sun Microsystems*, http://developer.java.sun.com/developer/onlineTraining/java3d.

Taub E., 2001, "Small Worlds to Create Bold, New Ones", *New York Times, March 1, 2001*, p. D7.

Turner A., Chapman D., Penn A., 1999, "Sketching a virtual environment: modelling using line-drawing interpretation", *Proceedings of ACM Symposium on Virtual Reality Software and Technology*, p.155-161.

VoloView, http://www3.autodesk.com/adsk/index/0,,224835-123112,00.html

Weborganic, http://www.weborganic.com/

Zeleznik R. C., Herdon K.P., Hughes J.F., 1996, "SKETCH: An Interface for Sketching 3D Scenes", *Computer Graphics, Proceedings of SIGGRAPH'96, Annual Conference Series*, p.167-170.

Graphics Interpreter of Design Actions

the GIDA system of diagram sorting and analysis

Ellen Yi-Luen Do
Design Machine Group, Department of Architecture, University of Washington, Seattle, WA 98195-5720, USA

Key words: diagrams, design action, graphics interpreter, design thinking, diagram sorting

Abstract: This paper describes the background and research for a diagram sorting and analysis system called GIDA – Graphics Interpreter of Design Actions. The GIDA system explores how spatial relations between graphic symbols in a design drawing may be extracted and the transformations of the graphic elements between different drawings in a design process may be tracked.

1. SCENARIO

Imagine one day you wake up to find that you do not remember who you are and what you do. A recent accident has injured your brain but left your motor skills and perception skills intact. You learn that you used to be an architect and designed many houses. Of most significance, you had been involved with a mental exploration, a house design for a single residence for the past ten years. All the design drawings are archived in paper format as well as scanned images stored on half a dozen CD-ROMs.

To help yourself get back your edge and to relearn how to be a designer, you decided to examine closely all these drawings to discover the relationships between the drawings, and to learn how one designer works, in this case, how you used to design. Your personal notes showed that the design was inspired by Le Corbusier's 'five points of architecture' principle and the idea of 'place within a place'. Everything seems easy, you have abundant drawings to examine, except for one problem: these drawings have no date or time information and therefore no sequential reference can be made.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 271–284.
© 2001 *Kluwer Academic Publishers.*

Now, look at Figure 1. Here are the drawings you pulled out from the archive and arranged them side by side on the table. You told yourself, this is easy: they are all projections of the same building: isometric, plan, section and elevation (assuming you remember what those different drawing types mean). Look again. Obviously they belong to the same family of the design drawing for the residential house because they share several design elements (Thick walls, hood, bridge, balcony, columns, chimney box and pipe). However, upon a closer look, there seem to be variations among the drawings. Some design elements were moved, rotated or height and width adjusted. This is fun, you think. If you can learn the transformations between design elements, and procedure of design moves, then you would learn how to design by following certain procedures.

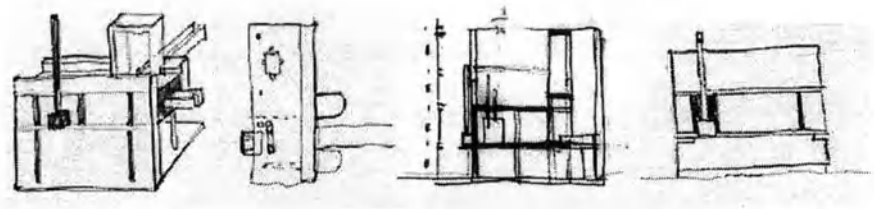


Figure 1. Various drawings of a pavilion house design

This is no fairy tale, nor do we wish anyone to be a victim of this nightmare. We are interested in developing computational tools to help us learn how architects design: specifically, the ways or arrangements, and transformations of design elements. Our previous work (Do, Gross et al. 2000) investigated the transformations of design elements in a series of drawings. We looked at the boundaries of the graphics space of these design actions, extracted and defined the rules that we could identify in the actions and products that it encompasses. The previous work suggested that a computational system might be built to emulate and analyse these actions.

In the following sections, we first present the background of this investigation of procedural knowledge in design process. Then section three examines related work on design cognition, including our previous work on spatial reasoning and a coding scheme. Section four describes the implementation of a graphics sorting and analysis system called GIDA explaining system architecture and functionality. Finally, we discuss some issues of analysing design drawings, and future research directions.

2. BACKGROUND

Recently we have seen a growing interest among cognitive scientists and design studies researchers in studying design drawings and diagrammatic reasoning. Researchers use methods such as think-aloud protocols (Akin 1978; Akin 1986; Akin and Lin 1995; Cross, Christiaans et al. 1996), retrospective analysis of design behaviour (Suwa and Tversky 1996), introspection (Galle and Kovács 1992) and analysis of design products (Schön 1983; Porter 1988) to study the role of drawing in design.

Drawing provides representations for problem solving, idea generation and exploration of design alternatives. Surprisingly, although drawing is the subject of this research studies have mostly focused on the verbal transcripts instead of the graphical representations that designers produce. Instead, we focus on design drawings themselves and propose a computational system to help identify and analyse connections between different design drawings.

In this paper, we report on the functionality and the implementation of a computer based graphical interpreter. We describe how this diagram sorting and analysis system can help design researchers to explore the space of graphic actions in which design intentions are expressed. We conclude with a discussion of future research directions.

3. RELATED WORK

3.1 Drawing and Design Thinking

One popular model of design studies draws a parallel of design with problem solving and views design as information processing (Newell and Simon 1963; Newell and Simon 1972). Design studies researchers and cognitive scientists use protocol analysis (Eastman 1968, Akin, 1978 #1580; Akin 1986) to study design as a process of problem formulation and solution generation. Transformations (links) of different states (nodes) are illustrated using Problem Behaviour Graph (PBG) notation. Chan's use of schemas (Rumelhart and Ortony 1977; Rumelhart 1980) represents domain-specific knowledge as design constraints and associated rules in memory (Chan 1990). Moran proposed that design (Moran 1970) consists of memory, representation conventions, interpreted problems, and design strategies. He argues that designer use many kinds of representations, or "languages" to express the state of the problem. The information processing model of design is also extended by researchers (Oxman and Oxman 1992) to include model-based refinements and case-based adaptations.

Design researchers also analyse the modalities of design actions. Schön argued that designers develop rules to guide their own thinking process (Schön 1988). He defined design as an act of 'reflection-in-action' (Schön 1983). Designers 'see' and then 'move' (Schön and Wiggins 1992) their design objects—through the act of drawing. Goldschmidt further suggests drawing is 'interactive imagery' (Goldschmidt 1991) and that designers use design drawing to perform 'seeing as' and 'seeing that' reasoning modalities (Goldschmidt 1989; Goldschmidt 1991). She argues that design, or design image transformation, is a systematic dialectic with oscillation of arguments carried out through the act of sketching.

Besides 'think-aloud' protocols, another type of studies uses introspective, retrospective or speculative knowledge. Galle and Kovács (Galle and Kovács 1992) argue that an introspective record provides ample time for reflection and eliminates the need to rely on either an 'information processing model' or other type of assumption for analysis. They argue that introspection is a useful supplement to either protocol or interview studies that often happen in a short period of time. Suwa and Tversky applied retrospective reports of design sessions (Suwa and Tversky 1996) to study designers' perceptual processes in observing their own sketches. They cited Dorst and Cross's review paper (Dorst and Cross 1995) to argue that think-aloud protocols has the disadvantage that concurrent verbalisation and behaviour could cause side effects or account for incomplete activities.

Porter conducts a "thought-experiment" using speculative re-construction of design process (Porter 1988) to account for the underlying logic of designing. He presents a chain of reasoning—arguments about how the design might have evolved—for two cases, an existing plaza and a building design from conception to their present state. Porter argues that a skilled designer or observer will discover design characteristics from its environmental settings—the appreciation of a site and his own reasoning of design principles. Therefore, a 'replication' of design process is a form of inquiry for design teaching and for revealing the implications of computer tools.

Architectural historians share this notion of relationships between design and its drawing. Hewitt argues that the history of architectural drawing is "a medium of thought" (Hewitt 1985). He argues that an 'idea sketch' consists of "personal and intuitive, or may be based on clearly defined methodologies or programs of instruction." This 'conception' of design is "a triad of interrelated operations–thinking, seeing, and drawing." Interestingly, a recent study by Akin and Lin echo this argument (Akin and Lin 1995). Their design experiment involved subjects reproducing a drawing from a printed transcript, and predicting the verbal data from a video of the design drawing process with the audio track suppressed. They conclude that novel design

decisions usually occurred when the designer was in a "triple mode period" of drawing, thinking and examining.

3.2 Related work on graphic systems

Many design studies research and computer aided design projects have attempted to provide better support for classifying graphics representation, automatic generation of graphic shapes, and recognition of building elements. For example, Electronic Cocktail Napkin (Gross 1994; Gross 1996; Gross and Do 1996) takes freehand sketching input from a digitising tablet and stylus and recognises shapes and diagram configurations. Each drawing mark has time stamp, pressure and speed information. Users of the system can define different drawing symbols with single or multiple strokes, as well as combinations of symbols to form diagrams. The Napkin project also has a replay function that can display drawing marks stroke-by-stroke in an animated sequence. The Design Amanuensis (Gross, Do et al. 2001) is a system that records voice input in parallel with recording drawing actions. Any segment of the design session (i.e., audio or the automatically recognised transcripts) can be later played back along with the drawing sequence. Computational Sketch Analysis (CSA) system (McFadzean, Cross et al. 1999) uses video cameras to record designers' drawing and physical activities. The graphical notational activities can be later replayed and used to build structures of the notations according to a predefined schema. HyperSketch I & II (McCall, Johnson et al. 1997; Gross, Do et al. 1998) record design drawings using a digitising tablet and generates a relationship coding (e.g., 'trace from' and 'design alternative') of designers' freehand drawings. It represents and links the drawings as nodes in a hyper document graph.

In addition to the research projects on recording design drawings, there are interesting efforts to identify graphic units in design drawing and automated recognition and reasoning of shapes. For example, Wang in his "Ways of Arrangement" presented a coding scheme (Wang 1987) focusing on the spatial relationships between objects (e.g., abut, adjacent). Our scheme (Do, Gross et al. 2000) on the other hand, focuses on the transformations of objects (design elements or drawings) among different states (e.g., staircase moved from east to west, wall height reduced). Achten's Generic Representations (Achten 1997) surveys architectural drawings to analyse the relations between graphic marks and design decisions. He classifies all design representations into one or a combination of three basic themes: 'shape', 'structure' and 'system'. For example, 'simple contour' and 'specified form' concerns the theme of 'shape'.

Likewise, the types of generic representation such as 'modular field', 'schematic axial system' and 'proportion system' all dealing with the theme of 'the structure underlying shapes'.

Goel's examination of designers in action reveals that the structure of drawing helps facilitate cognitive transformations (Goel 1995; Goel 1999). He argues that design process corresponds to ill- and well-structured representations and therefore non-notational and notational representations are both used. Specifically, the study of an experienced architect with right brain injury shows that drawing skills and the ability to design are ruled by different parts of the brain. In this example, even though the patient's drawing skills and explicit architectural knowledge base was intact, he could not make as many vertical and lateral transformations as a an uninjured architect. (Lateral transformation means the ability to generate design options between plan and sections, and variations of design of the same projections, i.e., a plan or sectional view).

Another direction of research is automatic recognition and reasoning of design drawings. For example, a computer vision technique was used to automate plan recognition (Koutamanis and Mitossi 1993). Design floor plans are defined as a configuration of architecture elements (i.e. walls, windows, and doors) and were used as templates for optical character recognition (OCR). With automated recognition of building elements and spatial subdivision, further analysis such as topological description can then be deducted. Gero, et al. develop a series of qualitative representation encoding scheme to attempt reasoning about two-dimensional and three-dimensional shapes (Gero and Park 1997; Gero and Damski 1999; Gero 1999). Their approach treats simple geometric shapes as exemplars of classes of shapes that can be represented by the vertex angle (A), length of edge (L) and curvature (K) which they call Q-codes. Using this coding scheme, drawing symbol configurations can then reveal the qualitative features such as repetition pattern or symmetry.

Our GIDA system is well situated in this family of design study research. The coding scheme that follows was developed from analysing an architect's drawing for a pavilion house design. The GIDA system uses the Electronic Cocktail Napkin as the graphic parsing and recognition engine. It extends the Napkin program with capabilities of comparing the state transformation between drawing symbol configurations, not just between drawing elements.

3.3 Our previous studies of design drawing

Any two drawings in a design project may share various properties: They may employ the same projection (plan, section, isometric), the same medium

(crayon, pencil and pen), or exhibit the same design intentions (circulation or views). They may describe the same elements (bridge, columns, and strip windows) in different configurations. They may be constructed from the same view angle.

In a previous study (Do, Gross et al. 2000) we looked at an architect's collection of over a hundred drawings for a residential house design and developed a coding scheme to classify these drawings. The scheme codes properties of the drawings such as the elements depicted and projection type and view angles. The notation system focuses on state transformations of design elements from one drawing to another (e.g., stair moved from east to west, wall height reduced) as well as changes of view and projection type. The codes facilitate easier comparison and sorting of element types and operations. However, the amount of descriptive data—the number of types and fields associated with each drawing quickly becomes difficult to manage. Furthermore, it is hard to keep track of the sorted design elements and their source drawings. Therefore, we built the GIDA system to help manage and sort design drawings by their features and transformations.

4. GRAPHICAL INTERPRETATIONS

GIDA (Graphical Interpreter of Design Actions) is a graphic spreadsheet sorting program for researchers and designers to identify and analyse individual drawings and their symbol system, as well as the transformations of composite elements in a series of drawings. It was built to facilitate easy sorting of graphical design actions among different design drawings. Designers can use the GIDA system to encode design drawings (such as the notation system described above) by making diagrams of different graphic shapes (circle, line, triangle, etc.) and entity features (i.e., colour, orientation, etc.) by tracing a design drawing. A researcher can assign identifiers to each individual element drawn.

Currently, the GIDA interface only allows two-dimensional input. However, the three-dimensional internal representation of a building can be formed by linking different projections (e.g. top, front, and side views) together. Figure 2 below shows that a three-dimensional model of the building (with design elements) is constructed and serves as a base template for recognising and comparing graphic elements.

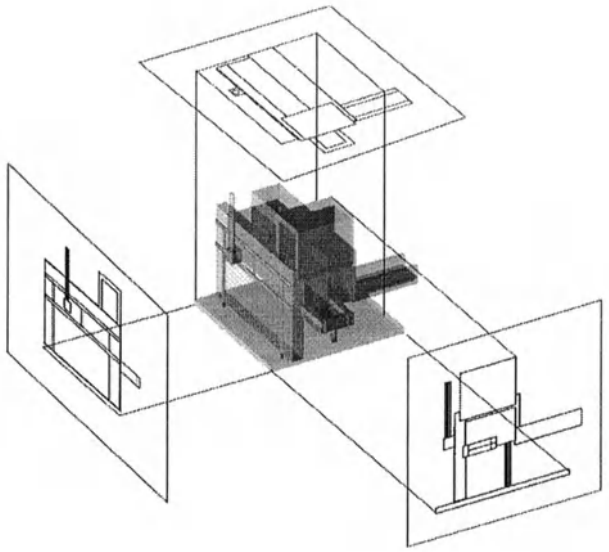


Figure 2. Two-dimensional projection representation of the three-dimensional building onto different planes (top, front, side).

The graphics engine of the program provides facilities to recognise simple shapes (circles, boxes, spiral) and configurations (overlapping boxes, crosses and T-junctions). The GIDA system then identifies the relative positions of elements in a drawing (on a 3x3 grid) and provides diagram-matching functions based on element types, element counts, and spatial relations. The transformations of elements among different drawings can also be identified and used as sorting criteria.

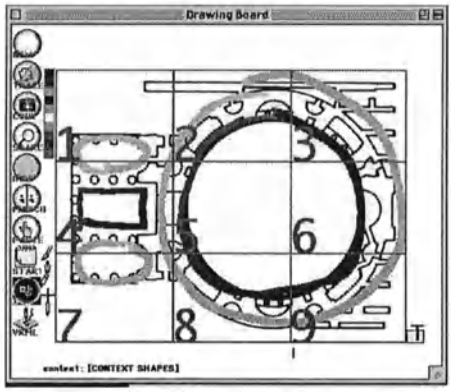


Figure 3. Picture underlay with analytical diagram and location identifier on a 3x3 grid

At the most basic level, GIDA can load any graphic file as an underlay base for diagram coding. After tracing and extracting the important design

elements, designer can call up GIDA's local position identifier (LPI) to analyse the locations of each graphic shape. The LPI adjusts the area for analysis to the smallest bounding box that contains all the drawing and then apply a 3x3 grid over the area. Then it generates the list of the cells through which each graphic element travels. Figure 3 shows an underlay of a church floor plan, the designer's analytical spatial diagram, and the top level LPI grid. For example, the top left circle of the church entrance corridor occupies cell 1 and 4, while the big circle representing the inner altar space occupies grid cells 2, 3, 5, 6, 8 and 9.

GIDA can also perform automatic feature extraction, or filtering. Figure 4 shows how a floor plan sketch (left) is extracted as a simple diagram configuration after filtering out overlapping lines and small drawing marks (right). The LPI can analyse both complex and simplified drawings and generate location lists for comparison. The goal is to identify the actions that transform a design, for example, object A was moved from location B to C. Therefore, extracting the diagram from the detailed design would significantly reduce the computation time and the generation of many uninteresting, low level object transformations.

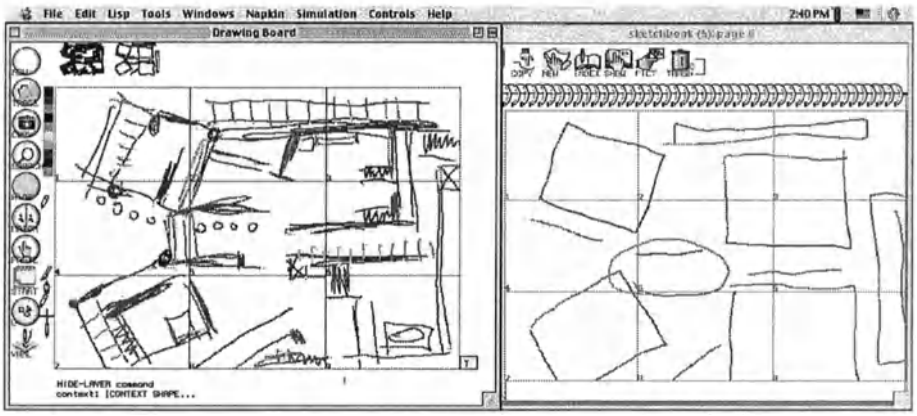


Figure 4. The filter function extracts sketch lines (left) to configuration of simple shapes (right).

Besides analysing the locations of graphic elements and the state change between any two drawings, GIDA has an analogy transfer function to apply a transformation of spatial relations from a source pair (A B) to a new drawing (C D). Figure 5 shows that given the example of a-window (big box containing a small circle) transformed to b-window (small circle moved from concentric to right-of the box), for any given c-window (big circle containing a small box), the system would generate a new resulting d-window (small box moved from concentric to right-of the big circle) through

this process of analogy transfer. This analogy transfer function would be useful to infer or predict possible design moves based on the design transformation operations found in the drawing analysis.

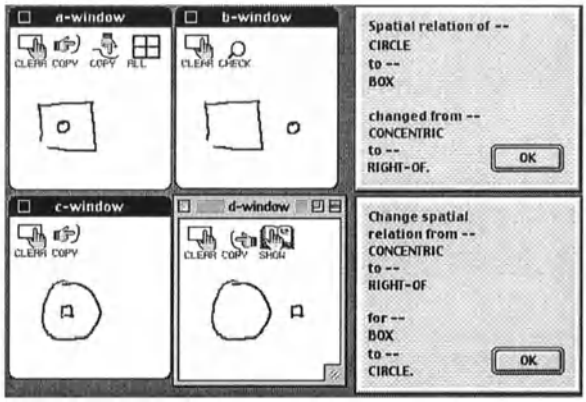


Figure 5. Analogy transfer, the transformation of spatial relationships from a-window to b-window is automatically applied to c-window to generate d-window.

Figure 6 left shows a pair of design drawings with frontal isometric views. Concerned about how element configurations are transformed between the two frontal façades, we could bring them in as underlay pictures and draw analytical diagrams on top of them. Figure 6 right shows the elevation diagrams traced over the pictures with underlay removed and LPI grid overlay. The GIDA system then generates a list of occupied cell numbers (drawn sequence) for each object, as shown in Table 1.

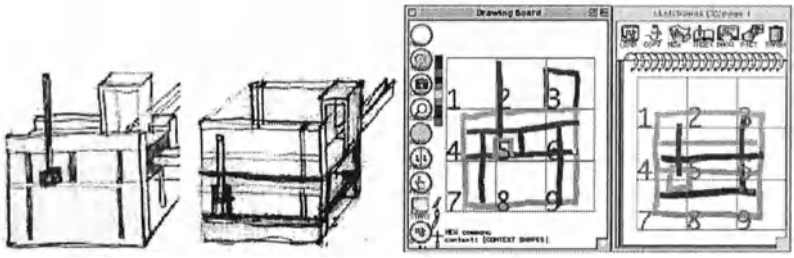


Figure 6. Design drawings of isometric views (left) and their diagrammatic analysis (right).

In this example, one drawing has nine elements and the other one has eight. Each element has a position in the global coordinate system and a list of the LPI cell sequence. Upon comparing the lists of the same element from different drawings, the transformation can then be inferred. For example, the Thick Wall's cell sequence list was changed from (7 4 5 6 9 8 7) to be (7 4 1 2 3 6 9 8 7). The sizes (bounding box) of this element in the two different

drawings are very close. Therefore, GIDA system inferred that the transformation for this element from the first drawing to the second one is a shifting up (addition of grid cells 1 2 3). The Hood element in drawing one is removed from drawing two. Likewise, the transformation of the Chimney Box is a moving down from (4 5 4) to (7 4 5 8 7).

Table 1. State of drawing elements represented as list of cell sequence.

Element	**Drawing #1**	**Drawing #2**
Thick Wall	(7 4 5 6 9 8 7)	(7 4 1 2 3 6 9 8 7)
Chimney Box	(4 5 4)	(7 4 5 8 7)
Chimney Pipe	(1 4)	(1 4)
V-window 1	(6 9)	(3 6 9)
V-window 2	(5 1)	
V-window 3	(4 7)	
Hood	(6 3 2)	
H-strip 1	(4 5 6)	(4 5 6)
H-strip 2	(4 5 6)	(4 5 6)
H-strip 3		(4 5 6)
H-strip 4		(7 8 9)

5. DISCUSSION AND FUTURE WORK

The GIDA system is a prototype, and a start for a larger research agenda on drawing and design action analysis. Many issues are worth exploring. First, although the current GIDA system can generate relations among any two graphics shapes or the transformation of a shape between any two drawings, the lists of relationship are too many and too detailed to be really useful. The obvious next step is to enable users to select a shape or a configuration and assign them specific naming classifications (e.g., parti, spatial themes or room names). Second, it's important to develop an easy to use interactive interface for selecting objects in different drawings for comparison. For example, an interface for designer to select the object in question in two drawings by picking or highlighting could be useful.

With this initial success of graphic action interpretations, we are musing over several interesting questions. Currently, we deal with transformations in only two-dimensional views. This is not surprising, considering that architects have been using orthogonal projections as a conventional way to communicate with others and to represent planning, for contracting construction, and to quickly generate alternatives. However, our building environment is three-dimensional and designer's design operations are also three-dimensional. What would be the coding scheme if we extended the three-by-three grid over to three-by-three-by-three, for a total of twenty-

seven cell locations? Or shall we change the 3x3 to be 2x2 so that we can always subdivide a cell in half to get a finer detail analysis and comparison? It would be interesting to apply the recursive decomposition to drawing. We will look into the literature and research on quadtrees and octrees in the field of computer graphics and image processing. Another question is: should the GIDA system be equipped with automatic three-dimensional geometric object generation abilities so that design transformations can be easily visualised? Digital Clay (Schweikardt and Gross 1998) and VR Sketchpad (Do 2001) are some systems that could be a useful extension for GIDA.

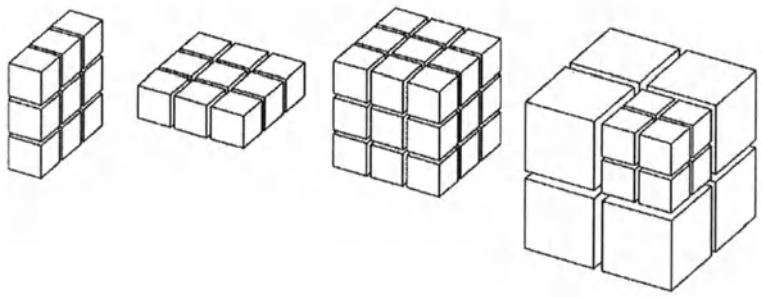


Figure 7. Variations of three-dimensional coding scheme using grid cells: slices of side and top views of 3x3, total consideration of a 3x3x3 cube grid system, and quadtrees with infinite subdivision by half to get finer comparisons.

In conclusion, the GIDA system is a proof-of-concept application and a vehicle for design researchers to investigate the graphical operations that designers use. We built the GIDA system to help identifying re-occurring pattern of design manipulations in a design process. Further functionality can be added to the system so that it could record and analyze design actions in a protocol analysis. The GIDA system is useful for analyzing and comparing design actions between drawings of the same project, different projects, or any drawings that can be illustrated as simple diagrammatic representations. It could also be a tool for students to learn about design by studying the sketches of others. Or it could be a tool to help designers reflect during the design process.

6. ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under Grant numbers IIS-96-19856 and IIS-0096138. The views contained in this material are those of the author and do not necessarily reflect the

views of the National Science Foundation. The author would like to thank Bennett Neiman for providing his pavilion house design drawings as experiment example for the project.

7. REFERENCES

Achten, H., 1997, *Generic Representations*. Eindhoven, Technische Universiteit.

Akin, O., 1978. *How Do Architects Design*. Artificial Intelligence and Pattern Recognition in Computer Aided Design, IFIP. E. J.-C. Latombe. New York, North-Holland Publishing: 65-104.

Akin, O., 1986, *Psychology of Architectural Design*. London, Pion.

Akin, O. and C. Lin, 1995, "Design Protocol data and novel design decisions", *Design Studies* 16 (#2, April), 211-236.

Chan, C.-S., 1990, "Cognitive Processes in Architectural Design Problem Solving", *Design Studies* 11 (2), 60-80.

Cross, N., H. Christiaans, et al., eds). 1996, *Analysing Design Activity*. New York, John Wiley & Sons.

Do, E. Y.-L., 2001. *VR Sketchpad: creating instant 3D worlds by sketching on a transparency window*. CAAD Futures 2001. B. d. Vries, J. P. v. Leeuwen and H. H. Achten. Eindhoven, Kluwer Academic Publishers.

Do, E. Y.-L., M. D. Gross, et al., 2000. *Intentions and Relations among Design Drawings*. Design Studies. G. Goldschimidt and W. Porter, Elsevier. 21: 483-503.

Dorst, K. and N. Cross, 1995, "Protocol Analysis as a Research Technique for Analysing Design Activity", *Design Engineering Technical Conferences* 2 (DE-83), 563-570.

Eastman, C. M., 1968. *On the Analysis of Intuitive Design*. Emerging Methods in Environmental Design and Planning. G. T. Moore. Cambridge, MIT Press: 21-37.

Galle, P. and L. B. Kovács, 1992, "Introspective observations of sketch design", *Design Studies* 13 (3), 229-272.

Gero, J. S. and J. C. Damski, 1999. *Feature-Based Qualitative Modeling of Objects*. Computers in Building: CAAD Futures '99. G. Augenbroe and C. Eastman. Dordrecht, Kluwer Academic: 309-320.

Gero, J. S. and S.-H. Park, 1997. *Computable Feature-Based Qualitative Modeling of Shape*. CAAD Futures 1997. R. Junge. Munich, Germany, Kluwer: 821=830.

Gero, S.-H. P. a. J. S., 1999. *Qualitative Representation and Reasoning about Shapes*. Visual and Spatial Reasoning in Design. J. S. Gero.

Goel, V., 1995, *Sketches of Thought*. Cambridge MA, MIT Press.

Goel, V., 1999. *Cognitive Role of Ill-Structures Representations in Preliminary Design*. Visual and Spatial Reasoning. J. S. Gero.

Goldschmidt, G., 1989, *Architectural sketching, seeing as and seeing that*, Unpublished manuscript submitted to the National Science Foundation.

Goldschmidt, G., 1991, "The Dialectics of Sketching", *Creativity Research Journal* v.4 (# 2), 123-143.

Goldschmidt, G., 1991. *Visual Clues: Tacit Information Processing via Sketching. 3rd Symposium on Systems Research, Information and Cybernetics*, Baden-Baden.

Gross, M. D., 1994. *The Fat Pencil, the Cocktail Napkin, and the Slide Library*. ACADIA '94. M. Fraser and A. Harfmann. St. Louis, MO: 103-113.

Gross, M. D., 1996, "The Electronic Cocktail Napkin - working with diagrams", *Design Studies* 17 (1), 53-69.

Gross, M. D. and E. Y.-L. Do, 1996. *Demonstrating the Electronic Cocktail Napkin: a paper-like interface for early design.* CHI 96, Conference on Human Factors in Computing Systems. Vancouver, British Columbia, Canada, ACM. Conference Companion: 5-6.

Gross, M. D., E. Y.-L. Do, et al., 2001. *The Design Amanuensis : an instrument for multi-modal design capture.* CAAD Futures 2001. B. d. Vries, J. P. v. Leeuwen and H. H. Achten. Eindhoven, Kluwer Academic Publishers.

Gross, M. D., E. Y.-L. Do, et al., 1998. *Collaboration and Coordination in Architectural Design: approaches to computer mediated team work.* Automation in Construction, special issue: TEAMCAD Workshop. C. Eastman. New York, Elsevier. 7: 465-473.

Hewitt, M., 1985, "Representational Forms and Modes of Conception: An Approach to the History of Architectural Drawing", *JAE* 39 (2), 2-9.

Koutamanis, A. and V. Mitossi, 1993. *Computer Vision in Architectural Design.* Design Studies, Butterworth-Heinemann Ltd. 14: 40-57.

McCall, R., E. Johnson, et al., 1997. *Hypersketching: Design as Creating a Graphical Hyperdocument.* CAAD Futures 97. R. Junge. Netherlands, Kluwer: 849-854.

McFadzean, J., N. Cross, et al., 1999. *Drawing and the Soliloquising of Design Suppositions.* Design Thinking Research Symposium. W. Porter and G. Goldschimidt. Cambridge, MA.

Moran, T. P., 1970. *A Model of a Multilingual Designer.* Emerging Methods in Environmental Design and Planning. G. T. Moore. Cambridge, MIT Press: 69-78.

Newell, A. and H. A. Simon, 1963. *GPS: a program that simulates human thought.* Computers and Thought. E. A. Feigenbaum and J. Feldman: 279-296.

Newell, A. and H. A. Simon, 1972, *Human Problem Solving.* Englewood Cliffs, NJ, Prentice Hall.

Oxman, R. and R. Oxman, 1992, "Refinement and adaptation in design cognition", *Design Studies* APR 01 v 13 (2), 117-134.

Porter, W., 1988, "Notes on the inner logic of designing: Two thought-experiments", *Design Studies* 9 (3 July), 169-180.

Rumelhart, D. E., 1980. *"Schemata": the building blocks of cognition.* Theoretical Issues in Reading Comprehension. R. J. Spiro, B. C. Bruce and W. F. Brewer. Hillsdale, NJ, Lawrence Erlbaum.

Rumelhart, D. E. and A. Ortony, 1977. *The Representation of Knowledge in Memory.* Schooling and the Acquisition of Knowledge. R. C. Anderson, R. J. Spiro and W. E. Nontague. Hillsdale, NJ, Lawrence Erlbaum.

Schön, D. A., 1983, *The reflective practitioner : how professionals think in action.* New York, Basic Books.

Schön, D. A., 1988, "Designing: Rules, Types and Worlds", *Design Studies* 9 (#3, July), 181-190.

Schön, D. A. and G. Wiggins, 1992, "Kinds of Seeing and their functions in designing", *Design Studies* 13 (#2), 135-156.

Schweikardt, E. and M. D. Gross, 1998. *Digital Clay: Deriving Digital Models from Freehand Sketches.* Digital Design Studios: Do Computers Make A Difference? ACADIA 98. T. Seebohm and S. V. Wyk. Quebec City, Canada, ACADIA: Association for Computer-Aided Design in Architecture: 202-211.

Suwa, M. and B. Tversky, 1996. *What Architects and Students See in Architectural Design Sketches: A Protocol Analysis.* First International Symposium on Descriptive Models of Design, Istanbul, Turkey.

Wang, M., 1987, *Ways of Arrangement: The Basic Operations of Form-making*, MIT.

Conceptual Design as HyperSketching

Theory and Java Prototype

Raymond McCall, Ekaterini Vlahos and Joshua Zabel
Sundance Group for Computing in Design and Planning
College of Architecture and Planning, University of Colorado

Key words: sketching, hypertext, hypermedia, conceptual design

Abstract: Hand-done design drawing still has a several advantages over current, CAD-based approaches to generating form, especially in the early stages of design. One advantage is the indeterminacy of hand drawing--i.e. its abstractness and ambiguity. Another is a non-destructive drawing process, where new drawings are created without modifying old ones. A third is designers' creation of large collections of inter-related drawings--i.e. graphical hyperdocuments. A fourth is the unobtrusive character of conventional drawing tools. These advantages might be taken as reasons for continuing to do early design on paper, but they also suggest ways in which CAD might be improved. We have created software prototypes that incorporate these features into a new type of CAD based on sketching with electronic pens on LCD tablets. One prototype, which we call HyperSketch II, simulates tracing paper in the sense that it enables the user to trace over previous drawings and to build stacks of traced over drawings. It also enables the user to create a hypermedia network in which the nodes are sketches and the links represent various relationships between sketches.

1. PROBLEM

CAD is increasingly used by architects for design, though it generally finds much greater use in the later stages of projects, especially in *design development.* In the conceptual and early schematic stages of design, however, we still see many architects preferring to design by sketching with pencil and pen on paper.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 285–297.
© 2001 *Kluwer Academic Publishers.*

Some people view the persistence of use of hand-done drawing in early design as merely a "generational thing"; they believe that when today's computer-literate children grow up they will design exclusively using tools like current CAD. We, on the other hand, believe this persistence reveals certain fundamental advantages of hand-done design drawing over current CAD. We do not, however, see this as an either-or choice between the computer and hand-done drawing. Instead, we argue that the advantageous features of traditional design drawing can and should be integrated into CAD software. In this article we describe software prototypes that move toward this goal by supporting sketching in design. Unlike work by most other researchers in this area, our emphasis is on providing support for the production, management and retrieval of large numbers of design sketches.

1.1 Differences that make a difference

Even a cursory examination of hand-done design drawings shows that they are strikingly different from the "drawings" or models created with CAD. In this paper, we take as our starting point the possibility that these obvious differences might represent important advantages of hand drawing for early design. In the following we describe a number of such differences and give our interpretation of their significance for CAD.

Difference 1: Hand-done design drawing is indeterminate in varying degrees, but current CAD models are highly determinate. By this we mean that hand-done design drawing is typically approximate, abstract, vague or ambiguous to some degree. It uses thick, wobbly or multiple lines; shapes are often rounded, lines only approximately parallel. CAD models, by contrast, use hard-line drawing and are typically based on precise dimensions and angles. Figure 1 shows an example of a hand-done design drawing.

It seems clear that the placement and shape of the lines in this example is only approximate and that this lack of precision is intentional and not the result of carelessness or incompetence. Conceptual sketches by the most famous architects reinforce this interpretation. As we see it, conceptual drawings such as the one shown here are meant to convey only approximate shape and location. Issues of precise dimensions and details are deliberately avoided and the "sketchy" appearance of such drawings is meant to convey that fact.

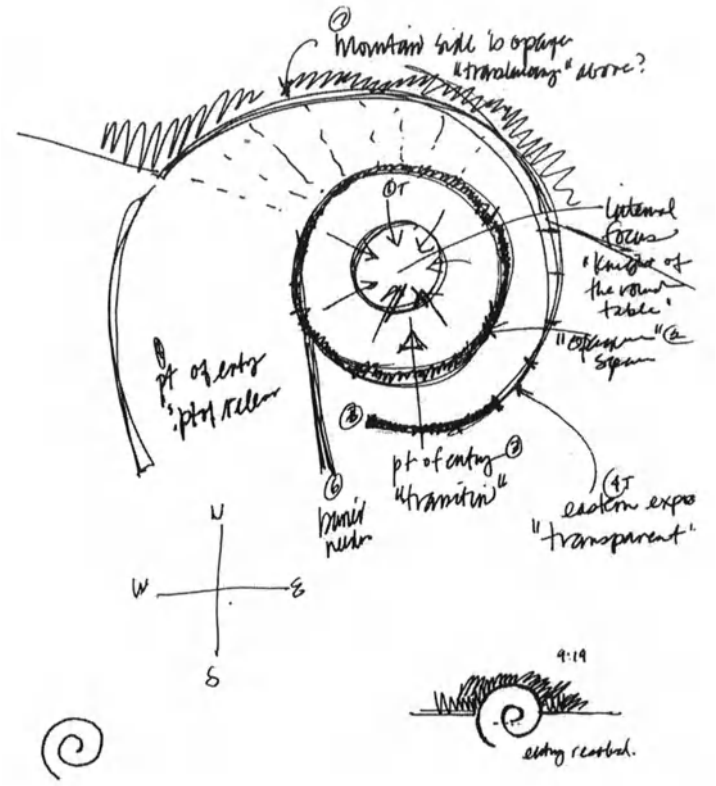


Figure 1. A conceptual sketch of a building with two "satellite" sketches in the lower corners.

That the indeterminacy of hand-done drawing is deliberate is further suggested by the fact that its degree varies systematically during design, generally becoming less indeterminate as design progresses. By being indeterminate, hand-done design focuses on larger issues of design while temporarily ignoring the many detailed issues that would arise in determinate drawing. Indeterminate drawing thus enables designers to use a *divide-and-conquer strategy* for attacking the complexity of architectural design. In other words, the sketchiness of the drawing is a means for decomposing the problem, for ignoring some issues to concentrate on others. In particular, issues dealing with precise location and shape are not yet being dealt with by the designer; they are a distraction at this point in the design process. They are likely to be tackled only after issues of general form and layout have been dealt with.

The implication here is that CAD—at least in its present form—does not provide as much opportunity as sketching for ignoring detailed issues. The hard-line character of CAD is a constant invitation to the designer to become distracted with such detailed issues as wall thickness, window dimensions and exact proportions of rooms. It also gives the viewer an unintended impression of precision.

Difference 2: Hand-done design is based on non-destructive drawing, but all current CAD is based on editing, an inherently destructive process. Once the initial version of a design solution—e.g., a model of a building—has been created with current CAD, creating the subsequent versions is accomplished by editing one version of this model to create the next version. This is a destructive process in that old states of the solution are destroyed—i.e. modified—to create new ones. Hand-done design, however, seldom involves modifying old drawings to create new ones. Instead, designers create new drawings from scratch or by tracing over previous ones. Destructive drawing, i.e., by erasure, generally has little or no significant role in early design.

Typically, the hand-done design drawings in a given project are highly heterogeneous. Different drawings represent many different levels of detail and different aspects of a building—including function, structure, appearance, circulation, lighting, user interactions, views, relation to site and surrounding context. Drawings show and compare alternatives at many different levels of aggregation and abstraction. As a consequence, most are not recognizable as being—even conceptually—edited versions of other drawings. Paper and pencil are thus much more than a "poor man's graphical editor," as is often supposed.

Especially important is the fact that use of non-destructive drawing preserves an *episodic history* of the project—i.e., a history in which graphical actions, such as the drawing of individual lines, are grouped into *episodes* corresponding to individual drawings and groups of drawings. Often a large sheet of paper is used as a way of grouping related drawings into still larger episodes—as shown in Figure 2. The episodic history of a project facilitates backtracking. It also enables evaluation of the design solution by examining its version history.

An editing process creates no episodic history. There is a history kept by the system to support *undo* functionality, but this is too fine-grained—at the level of individual operations on lines, shapes and groups of shapes—to be useful. Useful, high-level, cognitive groupings of such actions—such as drawings—are nowhere in evidence. CAD users often keep their own episodic history by saving intermediate stages in the drawing process in files and folders with names that they think up, but desktop CAD systems provide no explicit support for this.

Difference 3: Hand-done design creates a large collection of interrelated drawings but CAD creates only a single model. A large collections of drawings. Non-destructive drawing, by definition, produces multiple drawings, but this by itself does tell us how many drawings. Our studies suggest that a large number of drawings is typical. We observed one architect who created more than a thousand sketches in a two-week period. Many drawings by famous architects—such as the one shown in Figure 2—suggest the rapid production of many sketches during the conceptual design. Figure 3 shows one study of a professional designer who created 58 drawings in the first eight hours of a project. And in more that a hundred videotaped observations of students given a one-hour "sketch problem," we see them typically produce 6 to 25 drawings in that hour.

Current CAD, of course, is designed to support repeated editing of a single model. As mentioned above, designers can save multiple versions of their models in special files and folders that they create and name. But this approach is impractical when the number of drawings/versions goes into the hundreds—much less the thousands. Nor do current desktop CAD systems have either the database or version-control capabilities needed to manage drawings in such quantities. Furthermore, few CAD systems support creation of building representations in a minute or two, a regular occurrence in our studies of student design.

Interrelated drawings. In hand-done design, there are important relationships between drawings. Tracing is one indicator of such relationships. Some drawings represent alternatives to designs in other drawings. Some are done to resolve design problems that arise in doing other drawings; often the former are done in smaller scale on the side of the latter drawings—what we call *satellite drawings*. All of these represent crucial relationships between drawings. The groupings in the episodic history also represent relationships among drawings. In addition, design drawings are commonly annotated with related text, arrows, and numerical information. Thus—in addition to small "study" sketches—related diagrams, notes, tables and calculations are often found as "satellites" on the sides of larger drawings.

Some relationships among drawings are due to their sharing a common theme or exploring a common issue. Often such drawings are grouped on a single sheet of paper to show this relatedness—as for example in Figure 2.

Most relationships between drawings, however, are not explicitly documented. They are in the heads of designers during design. To identify such relationships we need to get information from designers while it is still fresh in their minds.

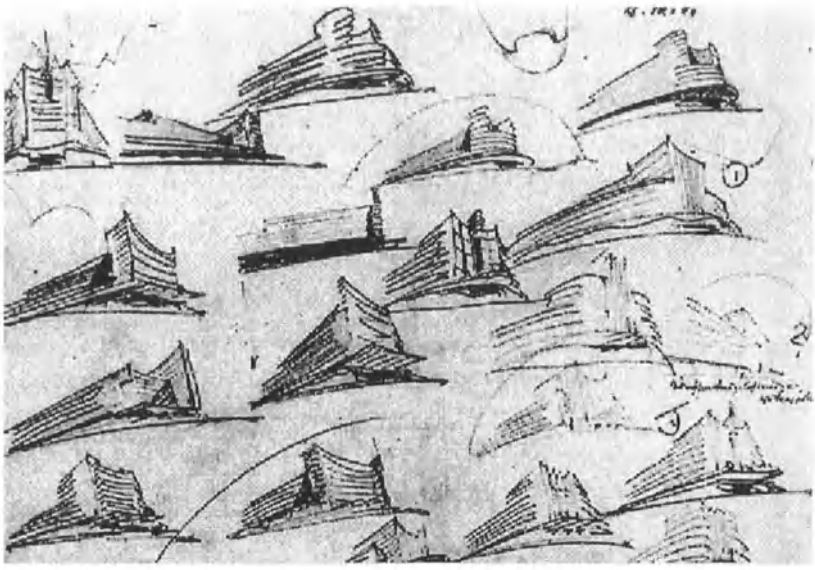


Figure 2. Sketches with a common theme on a single sheet of paper, by Erich Mendelsohn.

Figure 3 shows the results of an interview with a professional designer about the first eight hours of work on a project involving the redesign of a given house. The circles on the left edge of the figure represent the 58 individual drawings done in the eight-hour period. These are shown in the order in which the drawings were created, with the last at the bottom. (The top-most drawing is actually a site-plus-floor plan of the previous design of the house.) Note that there are a number of links directly between the individual drawings. For simplicity, we have represented all these links the same way. In reality there were six basic types of inter-drawing links that the designer felt it was important to distinguish. These included one drawing being 1) traced from (over) another drawing, 2) a further development of the design in another drawing, 3) a more abstract representation of the content of another drawing, 4) an alternative to the design in another drawing, 5) a part of another drawing and 6) a re-integration of a part into a larger whole. Some relationships combined several basic relationships—e.g., one drawing being a development of a part of another.

The small squares immediately to the right of the circles in Figure 3 represent sheets of paper. In this project it turned out that in every case where several drawings appeared on a single sheet of paper, the designer was consciously using the sheet to group related drawings. The larger squares further to the right represent major themes that group sheets of

drawings. The large circle on the right hand side represents the project as a whole.

The software needed to display and navigate through linked collections of information is called *hypertext* or *hypermedia.* Current CAD systems have not been designed with the hypermedia and database capabilities needed to represent the relationships between drawings and to manage extensive networks of drawings. If we were to extend these systems to deal with large collections of drawings—as we are proposing—it would be important to record the significant relationships between these drawings. Without such relationships it would be difficult to retrieve and make sense of collections of hundreds—much less thousands—of drawings. And such collections appear from our studies to be common in architectural projects.

2. RELATED AND PREVIOUS WORK ON COMPUTER-AIDED DESIGN SKETCHING

2.1 Related work

A number of researchers have created software to support sketching in design. These include Gross' extensive work with the "Electronic Cocktail Napkin" (Gross 1996), and the work of Tolba, Dorsey and McMillan (Tolba, Dorsey et al. 1999). Sketching has also been a recent theme at SIGGRAPH—e.g., (Ohen and Zelnik 1999) and (Igarashi, Matsuoka, et al. 1999). Some research has focused on sketch recognition—e.g., (Do and Gross 1995) and (Herot 1976); some on sketching as an interface for creating 3D models—e.g., (Zeleznik, Herndon et al. 1996) and (Schweikardt and Gross 1998).

Almost all of the attention of researchers has been on individual drawings. Relatively little has been written about *collections* of drawings and how they function in design. To our knowledge, no one other than ourselves has written about the way in which design drawings are implicitly or explicitly linked together to form a complex, graphical hyperdocument.

2.2 Our previous work

Since the 1970's we have used hypermedia in design (McCall 1979). In the mid-1980's we coupled hypermedia with CAD (McCall, Fischer, et al. 1989). Following this, we created PHIDIAS, a software system based on a concept we call *HyperCAD* (McCall et al. 1990)(McCall, Bennett, et al. 1994). HyperCAD integrates CAD graphics, knowledge-based computation

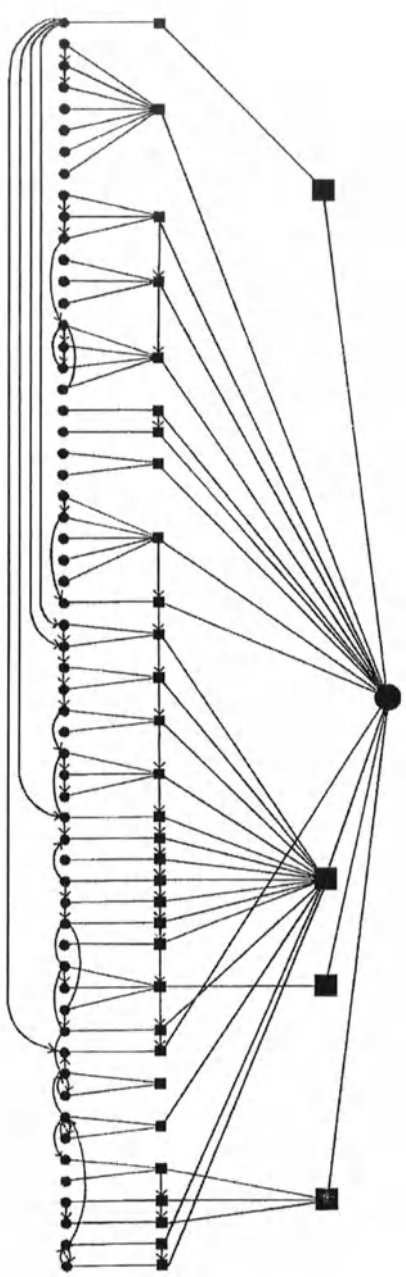


Figure 3. Graph of relationships among the 58 drawings done in the first eight hours of a house design project. All the relationships were explicitly identified by the designer as being important for an understanding of this collection of drawings. This graph is simplified in that different types of relationships are shown with the same type of arrows and without labels.

and hypermedia navigation using a unified computational substrate that manages databases of linked nodes.

In the 1990s we used HyperCAD to support of architectural sketching. The first results were 1) the prototype we call HyperSketch I and 2) an integration of hypersketching into PHIDIAS (McCall, Johnson, et al. 1996).

HyperSketch I was meant to do little more than simulate tracing paper on computer using a pen-based input on a touch sensitive LCD screen. It simulated the creation of multiple stacks of tracing paper but contained only the simplest drawing capabilities—i.e., one line thickness and no color—and very limited linking capabilities between drawings—i.e., primarily the linking of drawings based on their order in a stack of tracing paper. Tests of the system with computer-phobic architects (who were still in good supply the early 1990s) showed that it could immediately be used by designers with no CAD experience. But the HyperSketch I prototype did not attempt to show the advantages of using a computer for sketching.

We thought the most basic advantage of computer support would be the ability to manage large collections of linked drawings. In the mid-1990s, the best system at our disposal for managing large collections of linked data was the PHIDIAS system, whose hypermedia database engine we had been developing since 1981. We therefore integrated hypersketching capabilities into PHIDIAS (McCall, Johnson et al. 1997).

PHIDIAS was originally conceived as a stand-alone, desktop system. With the explosive spread of the World Wide Web we became convinced that the future of PHIDIAS' approach to hypermedia was to merge it with the Web. Online design offers crucial new opportunities for information access and collaboration, and these were precisely the goals that HyperCAD had originally been devised to achieve.

Initially we developed a prototype of PHIDIAS that could work over the Web (McCall 1999), but limitations of this prototype and the availability of new, software tools convinced us to scrap PHIDIAS and design a new, Internet-based HyperCAD architecture from scratch. This became the basis for the HyperSketching prototype described here.

3. THE HYPERSKETCH II PROTOTYPE

With the HyperSketch II system, as with its predecessors, designers create form by sketching with electronic pens on tablets. These system, however, is intended to go beyond pencil and paper alone by using the Internet to provide designers with "anytime, any place, any platform" access to both the software and the data management resources needed to create and retrieve large collections of interrelated sketches.

To provide access to sketching software and databases of sketches over the Internet, we opted for a three-tiered system architecture of the type has become standard and widespread in net-centric computing. The three tiers are as follows: 1) a client that runs on any Internet-enabled computer, 2) server-side software that combines a Web server with software for dealing with linked drawings, and 3) a back-end database management system. The middle tier mediates all communication between the client and the back end in such a way that the designer appears to have use of a database server that manages linked drawings—something that off-the-shelf database systems do not otherwise do. In reality, all the link management that connects the different drawings is accomplished by software in the middle tier. The linking in our case is hypermedia-type linking with links that are both labelled and typed and that are represented as first-class objects in the system. These links are also stored in the database. Such links can be followed in both directions in our HyperSketch II prototype. In other words, the linking mechanism is of the same basic type as that which provided the basis for PHIDIAS over the past decade of our research on that system.

For implementing the client, Java seemed an obvious choice. It combines platform independence with sophisticated Internet and interface functionality. Figure 4 shows the client in use. Note that the structure of drawings is shown on the right-hand side in the form of a *tree control*—of the same basic type seen in the Windows Explorer. This sort of interface is known in hypertext research as a *structure browser* and is generally regarded as an valuable aid to navigation in linked collections of information. It might seem at first that a tree is too limited to represent the sort of structure we saw in Figure 3, but trees—e.g., outlines—can represent arbitrary graph structures if any node is allowed to appear at more than one place in the tree. In addition to the tree representation, our client provides a thumbnail view of all sketches.

When the client starts, it provides the option of starting a new project or opening an existing one. If the user chooses the latter, the structure of the whole project is downloaded immediately; the individual drawings are only downloaded when displayed. This appears to be the only practical approach if hundreds of drawings—or more—are to be managed in a project.

We also opted for Java on the server. In particular, we used *servlets* to implement the software that mediates between client and database. Servlets run faster than CGI and are reputedly both more reliable and easier to manage. A servlet on this middle tier contains all the functionality for link management that makes the back-end database appear to the client like a hypermedia database—or *hyperbase*. In our prototype, the link-management servlet communicates with the back-end database using the JDBC (Java DataBase Connectivity) API (Application Programming Interface).

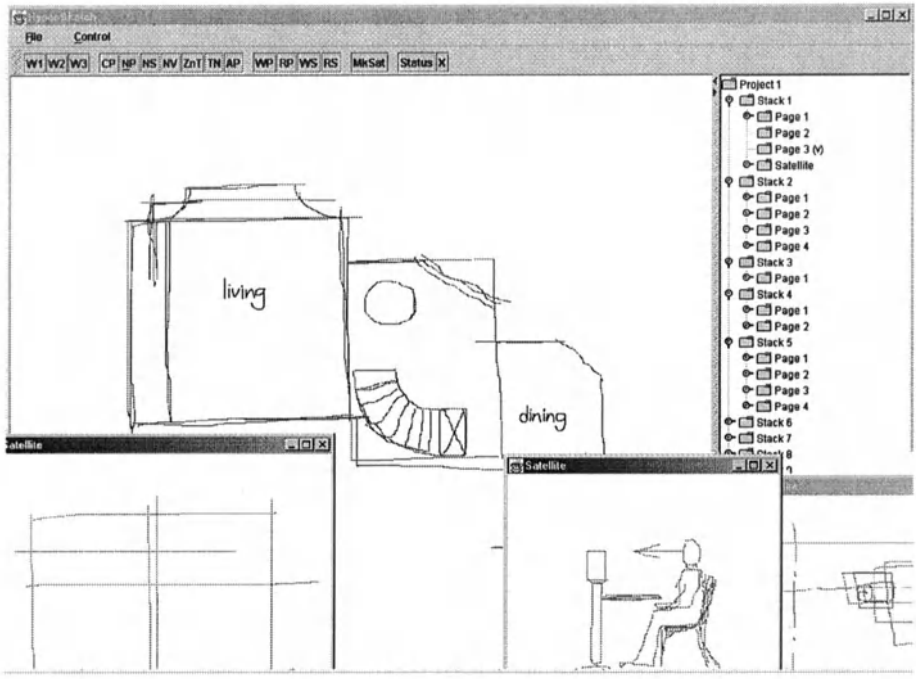


Figure 4. Screen image of the HyperSketch II prototype in use. Note the use (at the bottom of the screen) of separate windows for "satellite" drawings.

4. CONCLUSION AND FUTURE WORK

We have developed a new prototype, called HyperSketch II, that is aimed at showing how the Internet can aid design by enabling the creation, storage and retrieval of large collections of interrelated sketches from any Internet-enabled computer in the world. Our system is based on what we see as important advantages that hand-done drawing has for design, but we see our system's functionality as complementary to, rather than competitive with, conventional CAD. Hand-done design drawing and current CAD both have their strengths and weaknesses. Future systems can better aid design by providing integrated support for both.

It might seem that some of our functionality for creating multiple sketches could be achieved through clever use of existing, off-the-shelf software. But we suggest that such an approach is inelegant to the point of being impractical. As one software engineer we know is fond of saying, "Try it; you'll dislike it." It is precisely the business of CAD researchers to

devise explicit and well-designed support for design processes and not to be content with half-baked solutions that use off-the-shelf systems in ways for which they were not designed and for which they are ill suited.

Our system currently has been tested on desktop computers only. Soon we hope to use it on a variety of wireless devices, including tablet computers. This system enables designers to enjoy the benefits of sketching for the conceptual stages of design while also taking advantage of hypermedia functionality to manage large collections of related sketches.

Much remains to be done to exploit the full potential of networked computers for support of conceptual design sketching. Our future efforts will address the possibilities of using sketch recognition for informing designers and bringing knowledge-based computational aids into the earliest stages of design. We also hope to explore the use of sketching in collaboration. We plan to investigation additional navigational and retrieval aids. And, of course, we plan to build far more sophisticated drawing tools into future prototypes.

5. REFERENCES

Cohen, J.M., L. Markosian, R. C. Zeleznik, J.F. Hughes and R. Barzel, 1999, "An Interface for Sketching 3D Curves," ACM Symposium on Interactive 3D Graphics, pp. 17-22 (April 1999). ACM SIGGRAPH. Edited by Jessica Hodgins and James D. Foley. ISBN 1-58113-082-1 .

Do, E. and M.D. Gross, 1995, "Finding visual references by sketching," Proceedings, Association for Computer Aided Design in Architecture, 1995 National Conference, Seattle, 35-52.

Gross, M. D., 1996, "The Electronic Cocktail Napkin - computer support for working with diagrams," Design Studies. 17(1), 53-70.

Herot, C. F., 1976, "Graphical Input through Machine Recognition of Sketches.," in *SIGGRAPH '76 Conference Proceedings*, pages 97-102, July.

Igarashi, T, S. Matsuoka, and H. Tanaka, "Teddy: A Sketching interface for 3D freeform design," *SIGGRAPH 99*, 409-416.

McCall, R. "A Web-centric CAD System for Collaborative Design", in *Proceedings of the Eighth International Conference on Computer Aided Architectural Design Futures*, Atlanta, 7-8 June 1999, pp. 6.

McCall, R; E. Johnson, and M. Smith, "Design as Creation of a Graphical Hyperdocument," *CAAD Future '97 Conference*, R. Hunge (ed.), Kluwer Academic Publishers, 1997, pp. 849-854.

McCall, R., P. Bennett, and E. Johnson, "An Overview of the PHIDIAS II HyperCAD System," in *Reconnecting: ACADIA '94 (Proceedings of the 1994 Conference of The Association for Computer Aided Design in Architecture)*, A. Harfman (ed.), Association for Computer Aided Design in Architecture, 1994, pp. 63-74.

McCall, R.; P. Bennett, P. d'Oronzio, J. Ostwald, F. Shipman, and N. Wallace "PHIDIAS: Integrating CAD Graphics into Dynamic Hypertext," in *Hypertext: Concepts, Systems*

and Applications, Rizk, A.; Streitz, N. and Andre, J. (eds.), Cambridge University Press, 1990, pp. 152-165.

McCall, R.; A. Morch, and G. Fischer, "Supporting Reflection-in-Action in the Janus Design Environment" in *The Electronic Design Studio*, Mitchell, w.; McCullough, M. and Purcell, P. (eds.), MIT Press, 1990, pp. 247-259.

McCall, R., *On the Structure and Use of Systems in Design*, Dissertation, University of California, Berkeley, 1978, University Microfilms, 1979.

Schweikardt, E., and M.D. Gross, 1998, "Digital Clay: Deriving Digital Models from Freehand Sketches", *in Digital Design Studios: Do Computers Make A Difference? ACADIA 98*, T. Seebohm and S. V. Wyk, eds., ACADIA: Association for Computer-Aided Design in Architecture, Quebec City, Canada, 202-211

Tolba, O., J. Dorsey, and L. McMillan, "Sketching with Projective 2D Strokes, *UIST '99: Proceedings of the 12th Annual ACM Symposium on User Interface Software & Technology*, Asheville, North Carolina, November 7-10, 1999, CHI Letters, Vol 1, Issue 1, pp. 149-157.

Zeleznik, R.C., K.P. Herndon and J.F. Hughes. SKETCH: An Interface for Sketching 3D Scenes. In *SIGGRAPH '96 Conference Proceedings*, pages 163-170, August 1996.

Architectural Design Development through Multimedia Interaction

Yoshitaka Mishima, Peter John Szalapaj
University of Sheffield

Key words: analysis of form, dynamic interaction, conceptual models

Abstract: This paper describes the development of a multimedia system aimed at architects and architectural students for the purpose of helping them to understand the basic concepts of architectural *analysis*. Analytical features in the system that we have developed include many design-theoretic concepts such as massing, balance and circulation. Other concepts are more directly related to the built environment and include elements such as lighting, structure and construction. The system illustrates architectural analysis carried out on a range of building types *dynamically*, and allows users to navigate architectural analyses *interactively*. Users can learn about the differences between buildings and their corresponding analyses in a supportive non-linear learning process, and can explore building types depending upon their own interests or needs. The prototype system contains analyses of three British building projects. They show different types of architecture in order to demonstrate important design theoretic and environmental differences. *Conceptual models* in the system show important aspects of a particular analysis simply, and each analysis is additionally described with text, animations, video clips and interviews with architects (talking heads). Most of the models were generated by the use of architectural CAD software. Animation techniques were used to describe the analyses of buildings clearly and dynamically. Users can visualise how whole buildings were designed from an analytical point of view, and the system illustrates *design thinking* by showing dynamic presentations of analyses. Users can structure their own design learning processes through a series of interactions. These interactions are supported with flexible cross-referencing mechanisms implemented in Macromedia Director 8.0 exploiting Frame Markers, Event Handling, Navigation, and Buttons in the context of the object-oriented Lingo programming language. The navigation component of this system has a logical matrix structure reflecting the fact that analytical information is interrelated across building types, giving rise to vertical and horizontal patterns of access.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 299–314.
© 2001 *Kluwer Academic Publishers.*

The features of Director 8.0 can control this navigation in a flexible yet structured way. Users not only learn about analysis, but also how to present their own designs to the public through the use of different kinds of presentation techniques, particularly through the use of conceptual models. We intend that users can show their projects from their own analytical viewpoints instead of simply showing realistic images of final designs. Presentations can also be recorded in the system, and these can in turn be used as reference material for other users. This system is currently being developed further by storing presentations and translating them into different languages (e.g. Japanese) so that foreign users in other institutions can interact with these presentations. This system has been evaluated in the context of an undergraduate CAD course at the School of Architecture, University of Sheffield, UK. We are currently examining the usefulness of the system based upon an evaluation process, in addition to including more building types for future study.

1. INTRODUCTION

The essence of *analysis* in architecture is to understand what constitutes a building by categorising different parts in simplified detail rather than looking at whole buildings as complete objects. Analyses reveal the concepts of designs in particular categories, such as geometry, and structure [Clark, R. H. and Pause, M., 1985], for example. We are particularly aiming at the analysis of form in this paper [Baker, G., 1989]. Basic analysis of buildings using conceptual diagrams is an essential component of any multimedia system in order to describe analyses simply [Ching, F.D.K., 1996].

Multimedia CD-roms have to date been developed in various subject areas. Although teaching and learning through multimedia is an expanding area, it is also often one in which the technology of the medium undeservedly takes precedence over the pedagogical intent. We believe that this tendency should be reversed, at least in the case of applications for architectural education. There presently exists CAD software for designs, calculation programs for engineering, and software for presentation. Although there are multimedia CD-roms in architectural design, these are not always relevant for teaching and learning architecture, especially for the *analysis of form*. Our main objective is to introduce multimedia education in architecture to those who want to learn about the analysis of form.

2. ANALYSIS OF BUILDING FORM

The analysis of building form is concerned with understanding how buildings are designed by separating building types into particular parts and then investigating each of these in detail. Such an investigation may reveal how architects designed each part of the building, and how these parts are combined together in the final form. Analysis can show the design approaches of the architects, their techniques, use of materials, and so on. Therefore, it is necessary for students studying architecture to learn how to analyse buildings and to obtain ideas from analysis in order to develop their own design schemes. When students subsequently begin to work on their own design projects, they need to think in terms of their own design concepts, what kind of things they need to design, such as building type, space, elements, facilities, equipment, and functions. They need to have reasons for every part of their design. Analysing and understanding existing buildings may help them to develop their own design schemes by encouraging them to look at their own proposals in a more logical and rational manner.

Analysis of well-known buildings may allow architectural students to anticipate the properties of materials or the effects of natural light, for example. Analysis also encourages architectural students to uncover known design theories, techniques, or solutions to their problems. Architects are in turn influenced by other architects even though they may criticise other architects' works.

There are many types of analysis in architecture, such as the analysis of energy use, structural analysis, the analysis of form, the comparison of alternative material possibilities, and many other forms of environmental analyses. Our current research work focuses primarily on the *analysis of form*, i.e. those aspects of a design scheme that have spatial characteristics. We are concentrated *with how the form of buildings is developed* in both two and three dimensional ways. We are particularly aiming at the analysis of form because we anticipate that this will encourage students to develop their own design schemes in the *early stages* of projects before progressing to the details of finalised schemes.

3. ANALYSIS THROUGH MULTIMEDIA

In our own application of multimedia to the analysis of building form, we have considered this to be a relatively new medium for the understanding of architecture, and have therefore given some thought to the development of the following.

3.1 The Introduction of Pedagogical Interactive Multimedia Systems

Multimedia technologies have not been applied to architectural design in the early stages of studio projects in spite of the fact that a vast range of resources are being offered to us in schools of architecture. We want architecture students to be able to understand a range of building types such as those that they commonly encounter in studio work (e.g. libraries, schools, theatres, museums, galleries, housing, etc.), and to develop their own design schemes from the particular building types that they have been analysing. We are aiming to develop a computer-based, educational medium for students, which can be used in addition to reading books, papers, and journals. We believe that the development of multimedia environments with dynamic exploration and navigation will offer better support to students and architects for the analysis of buildings.

3.2 The Comparative Analysis of Different Architectural Viewpoints

Discussion about architecture helps students and architects to improve their own design developments whether they think by themselves or argue with other students and architects.

" Arguments give opposing views or a comparative analysis." [Sabater, J. C. and Gassull, A., 1992]

The hyperlinking mechanisms within multimedia environments make it possible to show the relationship between one type of analysis and another. They also enable users to compare analyses across different building types. Views and analyses give users the opportunity to see various new points of view. They can discuss these points by criticising or comparing different architectural solutions to particular building types. For instance, users can look at museums designed by several different architects in different styles and analyses. These different styles and analyses of particular building types encourage users to compare and contrast different analyses, to discuss different points of view, and these discussions in turn lead to the users' own design developments.

3.3 The Use of 'Conceptual' Models

Conceptual models contain analyses of each building. They are simplified diagrams in order to show specific analyses such as structure,

massing, circulation, etc. Arrows, for example, such as those used in **Figure 1**, can show the direction of movement of people in a building.

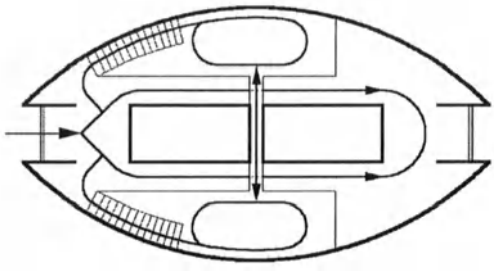


Figure 1. A conceptual model of circulation analysis

We believe that users can improve their abilities to analyse buildings by working more with conceptual rather than with realistic models, since conceptual diagrams can show the *essence* of particular building functions with clarity, and without the distraction of other elements. We believe that the use of animation in conjunction with two and three dimensional conceptual models will help students to visualise the forms of buildings in more realistic ways.

3.4 Relationship between Theory and Practice

Radford, in referring to Itten's basic course at the Bauhaus, emphasised the importance of:

> "the relationship between the theoretical and practical aspects" [Radford, A., 1992] , and stated:
>
> "The aims of a beginners' course on the making and interpretation of form, then encompass a combination of theory and practice in relation to design and the representation of designs" [op. cit.]

Our intention in the teaching of architectural analysis through the medium of multimedia is to encourage design students to:

> "relate theoretical ideas about the metaphor of design-as-language to the practice of design" [op. cit.]

We have already mentioned that we are incorporating *conceptual* models in our system according to the previous principle, and also that our system will lead students to discuss and develop their own design schemes. Then this makes it possible for students to realise their own design theories in terms of practical designs. This process enables students to think about the relationships between theory and practice. If students encounter problems

with either theories or particular practical problems associated with individual building types, such as method of construction, for example, they can then return to our system and look at particular issues in more detail. Structure, for example, is a primary concern in our system requiring special treatment. If students want to know the structure of a room with a large span during their own design process, they can open the *structure* section of our system and look at various large span structure examples. Design students can then learn about this particular element of buildings and develop their design theories accordingly.

3.5 Dynamic Interaction

A central feature of our own framework concerns the ways in which users can interact with the system allowing them to explore an architectural world effectively. We are aiming to provide an interactive multimedia teaching environment, in which it becomes possible for students to develop a greater awareness and understanding of a range of architectural types and design principles. This non-linear learning system allows users to pick up any one of the analyses they are interested in. The structure of this multimedia system is the next important thing to discuss.

4. MULTIMEDIA TECHNIQUES

"There is no right or wrong definition—it is a continuum of applications and technologies that allow for a wide range of experiences. In its most basic definition, multimedia can be thought of as applications that bring together multiple types of media: text, illustrations, photos, sounds, voice, animations, and video. A combination of three or more of these with some measure of user interactivity is usually thought of as **multimedia computing**." [Haykin, R., 1993]

Multimedia enhances traditional text-only computer interfaces and yields measurable benefit by gaining and holding attention as well as interest [Myren, B., 1994]. Multimedia information is a more effective way of understanding than through more conventional means because of the combination of sounds, voices, animations, and videos. When users themselves have interactive control of the process, they can be really become involved in the learning process. Multimedia stimulates the eyes, ears, fingertips, and, most importantly, the head. [op. cit.] An important feature of some, but by no means all, multimedia systems is the quality of dynamic interaction which enables users to navigate through information freely.

Within a free navigation system, hyperlinks are used to link pieces of information to each other.

4.1 Multimedia interaction

Multimedia is a medium within which users obtain information interactively.It is a non-linear rather than a linear information resource in which users can explore and navigate in order to obtain the information they want. They have choices of where to go according to what kind of information they need and how deeply they need to see the subject. The interface design will orient the user to the overall experience or message of the project. A transparent interface is one which is so subtle and quiet that users do not perceive an interface at all. In other words, coherent, easy to use *control and feedback* interfaces between users and computer need to be developed to let users navigate successfully through the system. [Cotton, B. and Oliver R., 1992]

4.2 Hyperlinks

In order to let users navigate through a multimedia system, hyperlinks connect information in the system depending upon what kind of links are needed. For example, buttons are used for browsing, opening images and video clips, and hyperlinked text makes connections to other related texts and to relevant images. Often, within multimedia environments such as web browsers, for example, these choices can have the effect of users being *lost in hyperspace*. In the implementation of our system called ADMIRE (an Architectural Design Multimedia Interaction Resource for Education), which will be described in subsequent sections, we attempt to preserve the flexibility of choice making within a flexible multimedia environment, whilst at the same time making the structure of the system itself explicit and transparent [Mishima, Y. and Szalapaj, P., 1999].

4.3 Macromedia Director

Macromedia Director is a sophisticated multimedia authoring software with an interface that lets users combine images, sound, video, and other media in any sequence and allows them to add interactive features through Lingo, the program's scripting language. Director is based on the metaphor of a theatrical production. A metaphor allows someone to understand and experience one kind of interaction in terms of another more familiar kind. Use of metaphors allows users to have a set of expectations that they can

apply to the multimedia environment. [Haykin, R., 1993] All the action takes place on the stage, and the cast appears on the stage as **sprites**, according to a timeline called the **score**, which tells cast members where to be and when to be there. A Director file is called a movie. Each movie, cast member, sprite and frame (thought of as a point in time) can also have its own script. Sprites are objects that span a range of frames. When you move sprites in the score, you move every instance of the sprite. The Director score provides a visual interface at every moment in time. According to Marc Canter, the founder and later director of VieoWorks:

> "Combining the time-line score and WYSIWYG (what you see is what you get) layout capabilities, you create a intuitive system for multimedia composition" [Allis, L., 1997]

Like a theatre production, a Director movie needs a **cast**. Each cast can have its own unique set of cast members which are media elements, such as graphics, sound, digital video, texts, or other Director movies.

5. STRUCTURE OF OUR SYSTEM

5.1 Information Types

The basic structure of our prototype ADMIRE system involves three building types (bridge, office, library) and three general categories of description. One of these categories contains general information about the buildings. Another category *Architect* includes the architect's biography and design influences. Location, site, plans, elevations, and sections are contained in the *Building* category. The most important category, however, which we especially concentrate on, is that of the *analysis* of the building types in terms of fifteen different analytical criteria which are typical criteria used in design-theoretical approaches [Clark, R. H. and Pause, M., 1985; Baker, G., 1989; Ching, F.D.K., 1996]. Many other types of analysis are possible, and other analyses will be included in further developments of this system (e.g. proportion).

We use various kinds of media, such as text, images, video clips, and talking heads. This information is structured in an x-y grid system in order to make the user-interface logically transparent to use. (**Figure 2**) The number of either analytical criteria or of building types can eventually be increased by extending the x-direction for buildings and the y-direction for the criteria.

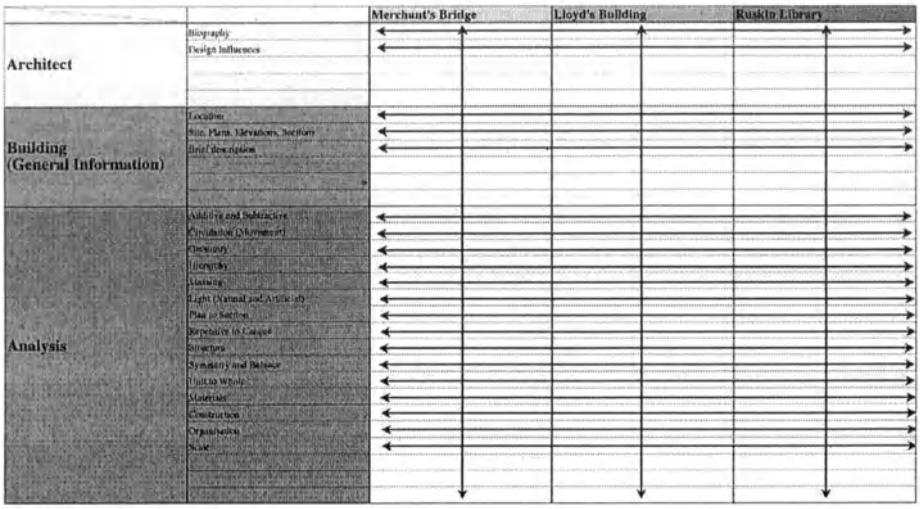


Figure 2. Grid system outline for navigation

5.2 Navigation

We expect that architectural students will use the system in such a way as to compare information across both building types and analytical criteria. There is no restriction on users to choose particular information in our system. Horizontal arrows in the chart of **Figure 2** shows the method of this comparison. Another way of learning is to see one building in terms of different analyses. Users can look at an analysis, visualise how the building is designed, and extract from the analysis essential components or ideas that can potentially be transferred onto their own design schemes. The vertical arrows in the chart illustrate the simple grid-based structure of navigation.

5.3 Interaction

We combined all the information types under the structure and navigation described in section **5.1** and **5.2**. Although the structure and its associated navigation system has a linear arrangement, the system gives free access to users so that they can start with whatever element they are interested in. However, once in a section, there are varieties of information such as animated diagrams, architects talking about how they designed, etc.

6. IMPLEMENTATION

6.1 Three British Buildings

We chose three British buildings for our system. The Merchants Bridge in Manchester designed by Mark Whitby, is a case study which encourages users of the system to focus on more engineering-oriented analyses such as *structure* and *balance*. The Lloyd's building in London by Richard Rogers represents an office building type. *Unit to Whole, Repetitive to Unique, and Plan to Section* are typical significant analyses of this building. The Ruskin Library at Lancaster University by Richard MacCormac is an example of library building type. The effects of *lighting* constitute an important analytical criterion for this building. Each building, therefore, has a different purpose and function, so that comparison of the analyses reveals different viewpoints.

6.2 Analysis of the buildings

We analysed each building in terms of the *conceptual models* described in **3.3. Figure 3** shows the geometrical analysis of the Ruskin Library.

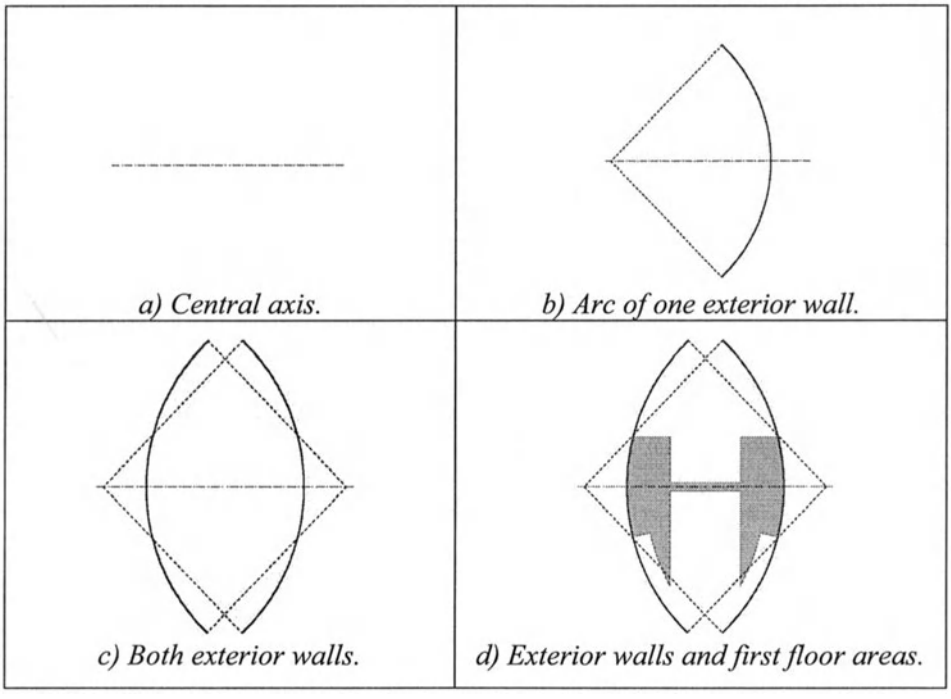


Figure 3. Geometrical analysis of the Ruskin Library

This diagram illustrates how the walls of the building are created from one single horizontal reference line. It also shows how the edges of the floor end at the points between the walls with lines for arcs. It is difficult to find this geometrical analysis in fully drawn plans. These simplified models clarify the method of design. The conceptual models enable us to show three dimensional analyses as well. The analysis of hierarchy is revealed with the model shown in **Figure 4**. Strength of colour (or shading) corresponds to the important areas in the building.

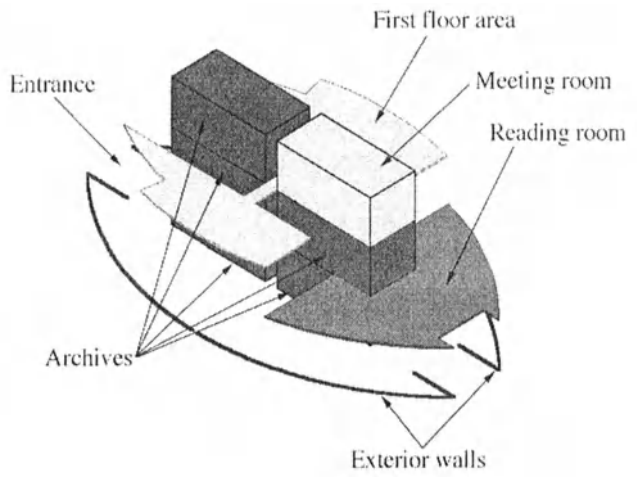


Figure 4. Hierarchy analysis of the Ruskin Library

6.3 Information Use

Each building type includes video clips associated with specific analytical criteria. The video clips that appear within individual analytical categories tend to give the user a more realistic understanding of the building type to complement the more conceptual types of analyses. In other words, they help users to map the theoretical ideas of the building onto actual built form. We also interviewed each architect and asked questions relating to the analytical criteria so that we can combine these two forms of presentation together to make forms of visual presentation more refined.

6.4 Dynamic interaction with the system

We use Macromedia Director 8.0 in order to accomplish the type of multimedia interaction we require. Director not only combines multimedia elements into a portable movie, but also supports them with Lingo,

Director's own interactive scripting language. The users interact with any of the media elements presented, and navigate through volumes of information. Users of our system have choices of information about the analysis of each building in the menu screen (**Figure 5**). This basic screen is reflected from our structure described in section **5**. An analysis of the building, which the users choose, is explained in the first page (**Figure 6**).

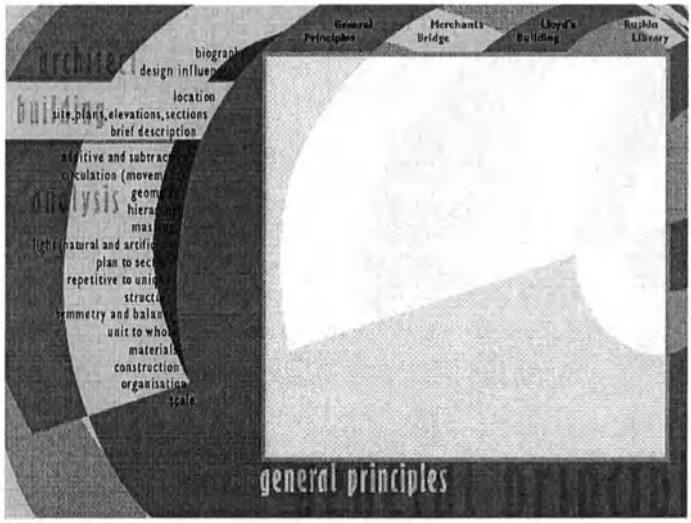


Figure 5. Basic ADMIRE system menu screen

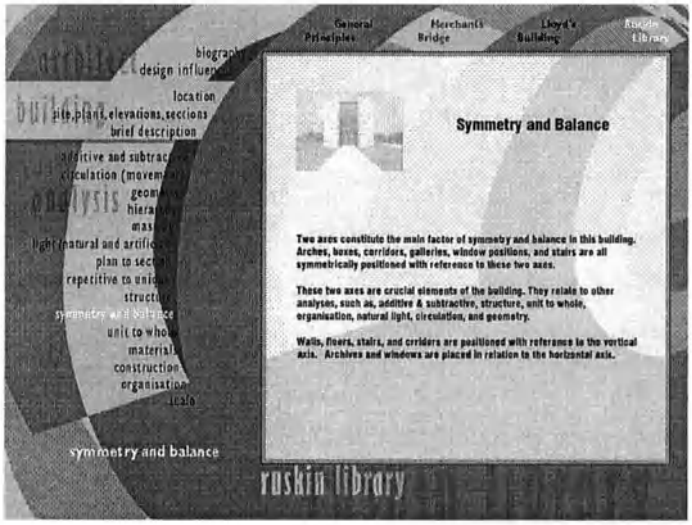


Figure 6. Introduction to the analysis of Symmetry and Balance in the Ruskin Library

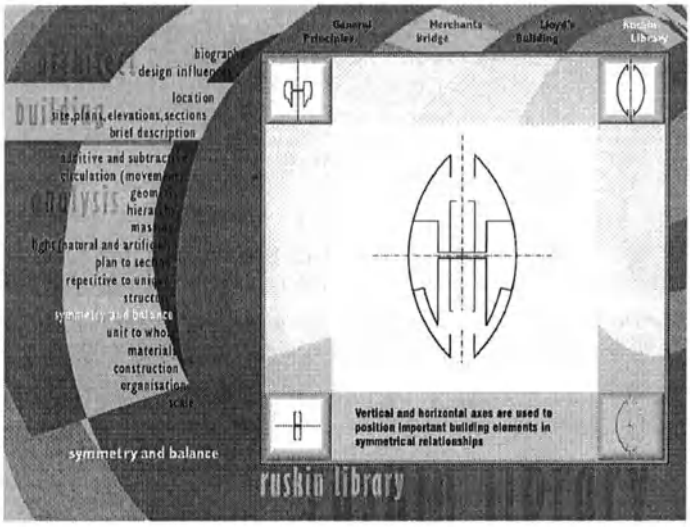


Figure 7. Conceptual diagram showing Symmetry in plan for the Ruskin Library

When they proceed to the next page, analytical conceptual models described in **3.3** are shown with text so that the users can see the building simply but also detailed (**Figure 7**). Realistic video clips of the building support the ideas of how this specific design scheme is applied to the final project (**Figure 8, 9**).

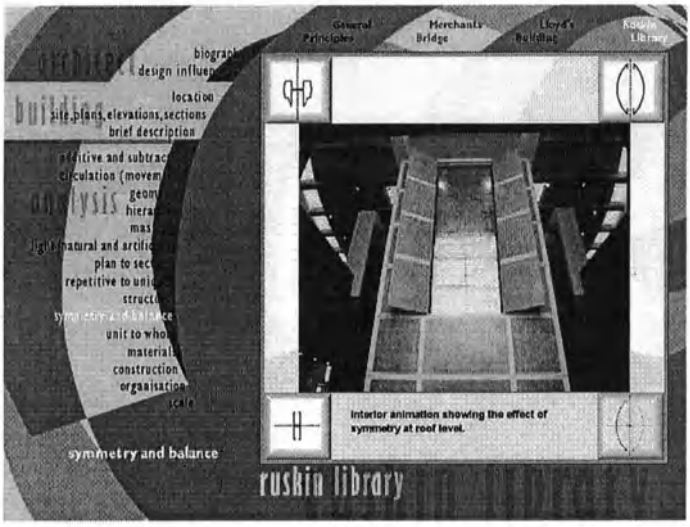


Figure 8. Realistic interior video clip illustrating wall symmetry at roof level

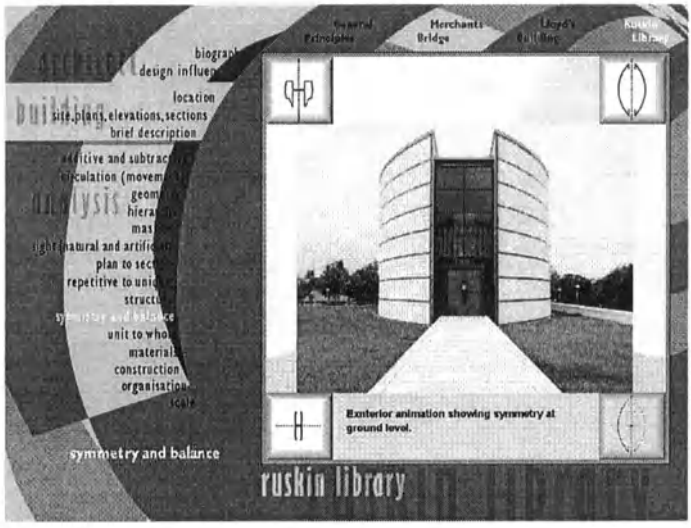


Figure 9. Realistic exterior video clip illustrating wall symmetry

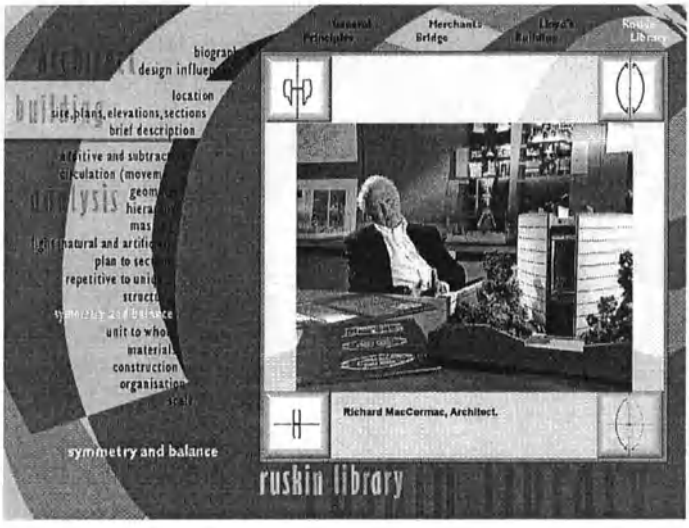


Figure 10. Talking head of Richard MacCormac describing symmetry

Listening to the architect talking about the design of the building reveals the range of ideas the architect had in the process of his or her design approach (**Figure 10**).

7. CONCLUSIONS

The advantages of interactive non-linear presentations for architectural education: Our multimedia system is based on the principles described in Section 4 which allow architectural students to build up a *context* for their own design schemes. These contexts will include various types of building, analysis, architect, etc. As the information increases within the context of a project, students have the opportunity to compare and contrast various new kinds of analyses. These analyses in turn encourage students to improve their own design theories.

User-centred learning: The system allows users to work at their own pace regarding the viewing of analytical descriptions. They are free to return to previously viewed analyses in order to reflect upon the information content. Since the ADMIRE system is available in CD-rom format, this work can be carried out outside of normal lecture and studio time.

Connecting the design studio with computer-based applications: *Conceptual* models show the main principles of buildings very simply. Our system clarifies these underlying principles, and the *conceptual* models can also be referred to by students for their own design presentations.

Bringing to life real buildings in real time: Students can also include their own design schemes in the system. These schemes can then be contrasted with the schemes of well-known designers in crit situations, for example. This system can ultimately become a collection of student work for schools of architecture. We are currently implementing a CD-rom teaching resource, in which students can interactively develop design ideas.

Evaluation of the system: We initially evaluated an earlier version of the ADMIRE system developed using HyperCard on a Macintosh computer in 1999. We found that second year students preferred less textual, and more visual information, as well as a more integrated system although the existing logical structure of the system was appropriate. We are presently compiling questionnaires that will elicit from second year architectural students their impressions of interacting with this new improved version of the ADMIRE system which currently contains three comprehensively analysed building types across fifteen analytical criteria. Our questions are aimed at getting reactions from students that indicate:
whether the system has presented them with understandable analytical information; the extent to which they are interested in investigating analytical possibilities; their criticisms of and observations of the user-interface of the system, and their suggestions and comments for the further development of the system.

In addition, we have implemented a function (using the Lingo scripting language) in Macromedia Director 8.0 to trace how the users explore and

navigate through the information in our system. This *tracing* or *history of use* mechanism allows us to monitor the amount of time that students spend within individual elements of the grid network, together with the connections they make themselves between the available analytical criteria. The responses of the questionnaires together with our investigation of exploration will be helpful for further development of the system, and the results of our evaluation will be published in future research proceedings.

We are also planning to carry out a long term evaluation lasting up to one year after students' use of the system, to investigate how the system has influenced students in the development of their own design schemes through studio project work. We expect to find a greater use of analytical diagrams, models and explanations not only as part of their final presentations, but also during the process of their design work.

8. ACKNOWLEDGEMENTS

We would like to thank members of the TV department at the University of Sheffield under the direction of Steve Collier for filming, editing, and production resources used in the development of the ADMIRE system.

9. REFERENCES

Allis, L., 1997, *Inside Macromedia Director 6 with Lingo,* New Riders Publishing, Indianapolis.

Baker, G. H., 1989, *Design strategies in architecture : an approach to the analysis of form,* Van Nostrand Reinhold, London.

Ching, F. D. K., 1996, *Architecture:Form, Space, and Order,* Van Nostrand Reinhold, New York.

Clark, R. H. and M. Pause, 1985: *Precedents in architecture*, Van Nostrand Reinhold, New York.

Cotton B. and R. Oliver, 1999, *Understanding Hypermedia*, Phaidon Press Ltd, London.

Gross, P., 1999, *Macromedia DIRECTOR 7 and LINGO AUTHORISED*, Macromedia Press, California.

Haykin, R., 1993, *Demistifying Multimedia*, Apple Computer, Inc. California.

Mishima, Y. and P. Szalapaj, 1999, "ADMIRE:an Architectural Design Multimedia Interaction Resource for Education", in: Brown, Knight, and Berridge (eds.) Proceedings of the 17th Conference on Education in Computer Aided Architectural Design in Europe, The University of Liverpool, Liverpool, p. 201-209.

Myren, B., 1994, *Multimedia: Making it work* second edition, Osbourne McGraw-Hill, California.

Radford, A., 1992: *Architecture as space and matter: Exercises in architectural composition and interpretation*, Computers in architecture , p. 33-40.

Sabater, J. C. and A. Gassull, 1992, *Learning from volume processing: working with a three-dimensional database*, Computers in architecture, p. 25-32.

Representation and Execution of Building Codes for Automated Code Checking

QZ Yang and Xiang Li
Manufacturing IT Division, Gintic Institute of Manufacturing Technology
Singapore

Key words: Object Orientation, Design Representation, Code Representation, Rule Execution, Automated Building Code Checking

Abstract: Computer-based representation and execution of building codes are investigated from the perspective of facilitating the operation of automated building code checking. The automated procedure proposed in this paper is based on the object-oriented (OO) representations for both building design and building code. These OO representation models fully support the execution of building compliance checking automatically. A prototype in Java on Windows NT has been developed to implement our approaches presented in this paper. Case studies are also conducted by using this prototype for automatic checking of building designs compliance with building codes under the "Technical Requirements for Household and Storey Shelters" of Singapore.

1. INTRODUCTION

Checking of building designs for compliance with building regulations and codes is typically a time-consuming, manual process that is highly dependent upon the professional checker's domain experience and skills. Automating the process is widely viewed as a complex and challengeable task to perform (Fenves et al., 1995 and Kiliccote, 1997) in computer-aided building design, due to:

- indeterminacy in some provisions of building codes,
- complexity of code structures and relationships,
- lack of unambiguous, consistent and complete representation models for building designs and building codes, and

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 315–329.
© 2001 *Kluwer Academic Publishers.*

– inefficiency in code execution.

Many efforts and attempts from research groups, trade associations, and software developers world-wide have been made to address these challenges in computer-aided building code compliance checking. Among them, the reference (Hakim and Garetta, 1993) reported the use of a description logic for representing both engineering designs and standards. Using their description comparison techniques, the representations of building designs and building codes can be compared for compliance checking. But this method did encounter the classification problems in standards processing with some description logic language. The typed feature structures proposed in(Carpenter, 1992) equipped with more efficient classification mechanisms, as well as other properties for support of the exploration of computational building design. The theory has been extended and implemented for building code checking through the representation comparison methodology (Chang, 1999 and Woodbury, et al, 2000). A hybrid prescriptive-performance-based approach for automated checking on disabled access provisions of America was proposed by Han (Han, et al, 1998). Based on this approach, software tools have been developed to support disabled access analysis. In Zarli's paper(Zarli and Debras, 1998), a system for conformity checking of building designs compliance with French national regulations for the disabled accessibility to buildings was presented. Test cases were reported using this Web-based code checking system.

Current international efforts for the development of standardised representation models of product information are from communities of STEP (Standard for the Exchange of Product model data) and IAI (International Alliance for Interoperability). Both are aimed at providing mechanisms for representation, exchange and sharing of computer-interpretable information about a product, a building in the case of AEC, throughout its project life cycle. And both are using a conceptual modelling language, EXPRESS(ISO, 1994), to describe product information structures and relationships. Several STEP application protocols (AP) are under development for AEC industry, including AP225 for structural building elements, AP228 for building services, and AP230 for steelworks. At present, these STEP building product models are still evolving and not reaching a mature level to have the support of major AEC CAD vendors. On the contrary, IAI's IFC (Industry Foundation Class) has been implemented in several AEC CAD packages which can consistently export valid IFC data files describing a building design. The current release of the IFC specification, IFC 2X (IAI, 2000), contains rich sets of class definitions to represent kinds of building objects. The hierarchy, properties and behaviours of these building objects are also specified in the model. To represent 3D CAD objects of building designs in an automated code checking process, IFC and its compliant information

representations may be the most suitable models in terms of their standardisation, unambiguity, consistence and completeness of description on building designs.

There are several software packages available for computer-aided building code checking. Usually, each package is implemented to only conform to local standards and practices of one country. BCAider (Sharpe and Oakes, 1995) from Australia is one of such packages. Using interactive communication with end users, the system provides services for automatic classification of building designs and compliance checking of building codes. Another package is BP-Expert (CORENET, 2000) developed in Singapore. This artificial intelligence based system checks architectural plans, 2D representation of building designs, for compliance to building codes of the relevant authorities in Singapore.

Although some other success implementations in the area of computer-aided building code checking can be found in literature, there are still quite a big gap between what the building industry requires for compliance checking and what the code checking researches actually offer. Specifically, more efforts and R&D activities are needed in these aspects:

- to develop common and sharable building data representation models and their matched building code representation models, and
- to implement both the data and code models on computer systems for the execution of building compliance checking automatically.

As one of such efforts to bridge the gap mentioned above, this paper proposes a new approach to represent and execute provisions of building codes for classified automatic building design conformance checking. The OO representation of building designs is described in Section 2. It is followed by the OO representation of building codes in Section 3. Section 4 illustrates the inference mechanism for the execution of building codes. A reference implementation of our approach is presented in Section 5. Case studies are conducted and discussed in this Section as well. Finally, conclusions and a remark on the further development of the reference implementation are sketched in Section 6.

2. REPRESENTATION OF BUILDING DESIGNS

The representations of 3D CAD building designs in an automated code checking process is described in this section, including a building conceptual representation model and a building object representation model. The development of an IFC-compliant building conceptual representation model called IM is discussed first. This is followed by the mapping of the model to

Java classes and the instantiation of the Java classes with IFC building design data from CAD to form the building object representation model. Lastly, the geometric reasoning of these Java objects is described.

2.1 Development of IM

The IFC methodology is applied to the development of IM. Although the existing IFC schemas have defined many universal AEC objects for the use of AEC applications in architectural design, cost estimation, building service design, construction, and facility management, the information modelled by IFC is not rich enough for the use of building code compliance checking in an automated process. To make more design information available directly for the process, IM, an IFC-based building conceptual representation model is developed. It supplements new building design objects and attributes into the existing IFC model to facilitate the automatic procedure.

IM is defined in EXPRESS. It is an extension of the standard IFC specifications. As a conceptual information model mainly for architectural design, IM consists of computer interpretable definitions for architectural design entities, semantic relationships and constraints on entities and their attributes. The other AEC entities, IFC hierarchy, and global rules governing the use of the entities in the existing IFC are still maintained in the IM.

To meet the requirements from the automated checking procedure and to reduce the workload for the later geometric reasoning, additional TYPE objects and some entity attributes distinguishing the use of these entities are specified in the IM and integrated with the existing IFC model. For example, to identify, more effectively, which SPACE is the relevant design object for a checking request on the clear area of a household shelter and its capacity for sheltering people, an extra attribute for the SPACE name has been added to the SPACE entity in the IM. Through the use of this attribute, the shelter SPACE object and its parameters relevant to this checking request can be searched and extracted more easily and quickly. Furthermore, any other design information, such as shape geometry, material behaviour, or surface characteristics pertaining to other interested building objects can also be retrieved effectively through this identified shelter SPACE object by object navigation with the IM hierarchy.

2.2 Java mapping of IM

As indicated previously, IM as a conceptual building model is defined in EXPRESS which does not contain any elements to support execution. Therefore, IM must be implemented in a programming environment for representing real building design data in CAD files.

Taking the advantage of computer-interpretable nature of the EXPRESS language, the process of mapping IM to its Java equivalent is handled automatically by an IM to Java translator. The tool converts the EXPRESS constructs in the IM specifications into respective representations in the target executable language Java. The structures and relationships of these resultant Java classes are consistent with those of the IM conceptual model.

In CAD process, building design data can be exported to IFC files. These IFC data files are then used to populate the Java classes into Java objects to form a building object model. It is an accurate representation of the original IFC data files for a CAD building design submission. Each Java object in this representation model can be accessed and analysed in a geometric reasoning process for the extraction of the design parameters used in building code checking. The processing flow of mapping from IM specifications to Java classes, and from CAD design data to Java objects is illustrated in Figure 1.

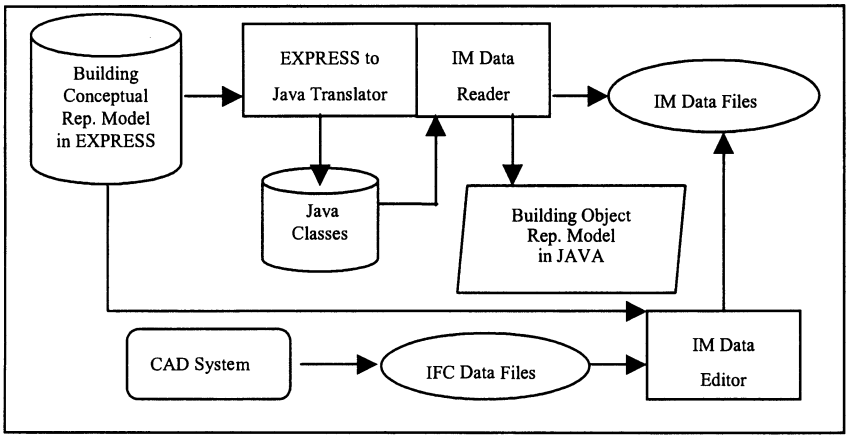


Figure 1. EXPRESS to JAVA processing flow

2.3 Geometric reasoning

Code checking requests from different authorities and agencies often ask for design parameters which are not directly available from the IM conceptual representation model. Even these parameters are specified in the conceptual model as attributes of design objects, they may have NULL values in the IM object representation model because:

- designers ignored the settings for these attribute values in CAD design process; or
- CAD packages exported incomplete IFC data files for a building design.

If any case mentioned above happens, these design parameters will not be available for compliance checking. So geometric analyses are needed to reason those derived parameters using valid available values in the IM object representation model.

Take a checking request to the size of household shelter as an example. The clear area of household shelter is an item asked for checking in our sample regulatory requirements extracted from the Reference (PWD, 1997). Although the SPACE area is defined as an attribute in the IFC, in many cases, the attribute value is still not specified during building CAD design process. As a result, no valid value for this attribute will be available for compliance checks. Geometric reasoning algorithms for the calculation of the SPACE area have to be developed to derive this attribute value based on known design parameters in the design model.

The more explicit properties are defined in the IM conceptual model and instantiated in the IM object model with complete IFC data files, the less complexity of geometric reasoning algorithms is. The efficiency of geometry processing is also depending on the model structures and object accessibility of both IM representation models.

3. REPRESENTAION OF BUILDING CODES

This section discusses the development of a rule representation model for building codes used in automated compliance checking. The development process includes three steps: building codes classification and abstraction, rule representation modelling, and knowledge base establishment.

3.1 Building codes classification and abstraction

To represent building codes properly in a building rule representation model, the codes should be classified first. A code classification scheme is specified in our approach as follows: First, the nature of a code is interpreted in the context of a domain. All objects related to the code are then identified. After this, the measurability of these code objects and the possible checking processes applicable to the code objects are evaluated. Based on analyses to the code, it is finally classified into indeterminate provisions or measurable provisions. Only these classified measurable provisions of codes are taken into account for automated building code checking. For example, such a code like "every household shelter should be designed to a specific peacetime usage" will not be considered in the checking process discussed here.

Codes abstraction applies only to the measurable provisions and is the next necessary step for the establishment of a rule representation model. Table 1 shows examples of how facts and rules are identified and extracted from the building codes under Reference (PWD, 1997). Through identifying relevant objects in a measurable provision, such as "total floor numbers and total unit numbers" listed in Table 1, the facts and rules are extracted. The facts are criterion values specified in building codes, while the rule objects are pairs of preconditions and consequences defined from the understanding to building codes. The facts may be "Null" if the codes do not specify any criterion data, but measurable (see Table 1 for an example). The rule objects extracted from sample regulatory requirements are also listed in the table.

Table 1. Examples of facts and rules extraction from building codes

No	Building Codes	Objects Identified	Facts Extracted	Rules Extracted
1	The household shelters (HS) on every storey shall be located one on top of the other, to form a vertical HS tower with continuous walls.	1) Total floor numbers 2) Total unit numbers 3) HS tower 4) HS(i^{th} floor, j^{th} unit) = (X_{ij}, Y_{ij}, Z_{ij}) [k]* 5) HS(criterion of jth HS Tower) = (X_{0j}, Y_{0j}, Z_{0j}) [k]*	Null	If (X_{ij}, Y_{ij}, Z_{ij}) [k] = = (X_{0j}, Y_{0j}, Z_{0j}) [k] then "Passed", else "Failed". k = 1,2,3,4.
2	The length to width ratio of the HS floor area shall always be less than 3:1.	1) Length of HS floor area 2) Width of HS floor area	3	If (length/width < 3), then "Passed", else "Failed".

* (X_{ij}, Y_{ij}, Z_{ij}) [ ] – a vertex coordinates array for a household shelter of jth unit on ith floor.
(X_{0j}, Y_{0j}, Z_{0j}) [ ] – a criterion vertex coordinates array for jth household shelter tower.

3.2 Rule representation modelling

The object orientation methodology is used for the development of a rule representation model. The rule modelling process consists of two aspects: identifying rule classes with their relevant attributes and establishing the relationships among these classes.

Identifying classes for rule objects goes through the categorisation of relevant objects into groups, such as space, door, or wall, and the definition of their sub-classes and attributes, such as clear area of HS space, thickness of a living room wall, or width of a kitchen door. Objects with similar attributes are classified into one class. This classification also depends on the requirements of a building code and its checking process. For example, in the Reference book (PWD, 1997), width and height of a door are used to

describe the dimension of the door. So the "Door" class should be defined with these two attributes.

Establishing the relationships among the classes of rule objects is a critical process in rule modelling. The design of a knowledge base is mainly depending on these relationships. For example, to check the first building code listed in Table 1, the class relationships between the "space of HS" class and "wall of HS" class should be established. That is because the checking process asks for relevant vertex coordinates of HS walls in order to identify the HS space location. These relationships have been defined in our rule representation model.

3.3 Knowledge base establishment

Unlike a traditional rule base generated by a rule editor, a knowledge base consisting of tabular forms can be created by any editing tools. The facts and values of rule preconditions are stored and maintained in the tables of the knowledge base. The following building codes are used to describe the knowledge base establishment: (GFA = Gross Floor Area)

If GFA <= 45 m^2, then minimum internal clear floor area of HS = 2.0 m^2.
If 45 < GFA <= 75 m^2, then minimum internal clear floor area of HS = 2.4 m^2.
If 75 < GFA <= 140 m^2, then minimum internal clear floor area of HS = 3.2 m^2.
If 140 < GFA then minimum internal clear floor area of HS = 4.0 m^2.

Based on these codes, a rule table can be created by using steps below:

Identifying code objects. The code object of "GFALower" is used for holding the preconditions of "lower bound of GFA of dwelling unit", while the "GFAUpper" for "upper bound of GFA of dwelling unit". "MinArea" is for the minimum internal clear floor area of HS.

Extracting values of preconditions. The values of 45, 75 and 140 in the codes are extracted as values of preconditions of GFA.

Defining facts. The values 2.0, 2.4, 3.2 and 4.0 in the codes are defined as the facts of "MinArea".

Creating representation table. The code objects of "GFALower", "GFAUpper" and "MinArea" are selected as the table fields, then the values of preconditions of GFA and facts of "MinArea" are inserted to the relevant rows and columns to form a table shown in Table 2.

Table 2. Tabular Representation Example

ID	GFALower	GFAUpper	MinArea	...
1	-	45	2.0	...
2	45	75	2.4	...
3	75	140	3.2	...
4	140	-	4.0	...

There are three main advantages for the use of a tabular representation. First, it is easier to express the rule facts and preconditions in a table than in programming. Second, it is more flexible to supplement the table with additional knowledge simply by adding more rows and columns. Third, it is convenient to maintain the knowledge base by traditional database software.

Once all tables needed are completed, a knowledge base can then be established using these tables and rule classes defined early. Use this approach, the knowledge base will be of such a schema with mirrored class structures from the IM representation models, which makes it easier to access required facts and to generate any active rule models from the knowledge base according to the checking criteria specified by a user.

4. INFERENCE OF RULE OBJECTS

In this section, the rule processing procedures are described first. It is followed by an introduction on the inference software environment we used. Finally the rule objects execution under this environment is discussed using a rule-firing example.

4.1 Rule processing procedures

The procedures of rule processing are shown in Figure 2.

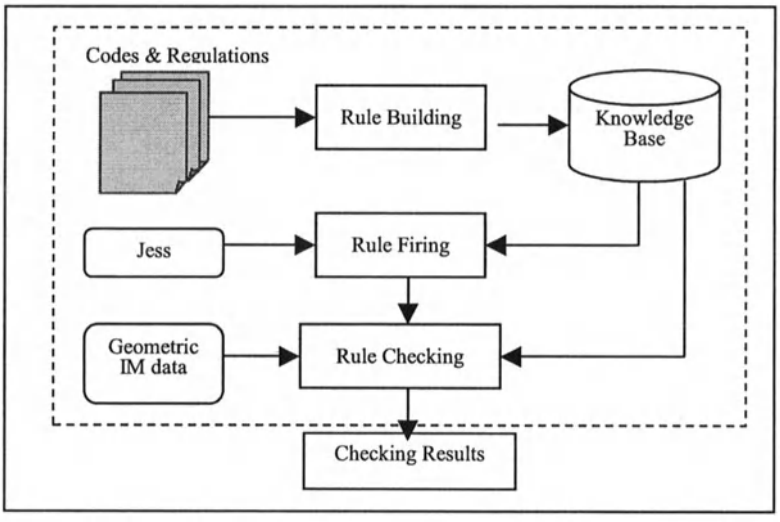


Figure 2. Procedures of rule processing

Rule Building. The procedures start from acquiring the building codes and regulations from the reference book(PWD, 1997) to build rules. It firstly extracts the code objects, facts and rule preconditions from the building codes, categorizes these objects into classes and identifies their relevant attributes. Relevant data are then inserted to tables to form a knowledge base. The knowledge base is defined to store two type of the values: facts (criterion data) and values of rule preconditions. The facts can be directly used to check geometric IM data, while the values of rule preconditions are used to initiate rule firing.

Rule Firing. A rule engine is used to automate the rule reasoning. The Java Expert System Shell (Jess) (Ernest, 2000) has been selected as the rule engine and the scripting environment for Java programming of rule objects. All the facts and rules which need to be fired are accessed by the Jess engine called Rete. The rule engine not only works for rule firing, but also fills out the facts from the knowledge base for building code checking. The inference principles of the rule engine with Jess are described in the next section.

Rule Checking. In the rule checking procedure, the real design parameters are automatically compared with the facts extracted from the building codes based on a checking request, so that checking results can be derived and sent out to the requester.

4.2 Execution of rule objects

Using an object-oriented method to implement a rule engine for automated codes checking is one of the principal focuses in our approach. Jess can satisfy all requirements specified in the approach. With Jess, we can embed Rete engine into our own rule programming in Java. For example, we may have the following rule (refer to Table 2):

"If $45m^2$ < GFA <= $75\ m^2$, then minimum internal clear floor area of HS = $2.4\ m^2$"

To process this knowledge, the rule is translated into Jess representations and accessed by "Store" and "Fetch" functions of Jess as follows:

```
Rete rete = new Rete();
rete.store("GFA"     , new Value(gfa  , RU.FLOAT));
rete.store("GFALower", new Value(lower, RU.FLOAT));
rete.store("GFAUpper", new Value(upper, RU.FLOAT));
rete.store("MinArea" , new Value(minarea, RU.FLOAT));
rete.executeCommand("(defrule startup" +
      " =>" +
      " (assert (input-1 (fetch GFA)))" + " (assert (input-2 (fetch MinArea))))");
```

Jess rules are defined as follows:

```
rete.executeCommand("(defrule rule1" +
    " (input-1 ?a&:(> ?a " + lower + "))" +
    " (input-1 ?a&:(<= ?a " + upper + "))" +
    " (input-2 ?c)" + " =>" +
    " (store RESULT (+ ?c 0 )))");
rete.reset(); rete.run();
if (rete.fetch("RESULT")== null) {
   System.out.print("Error: Wrong conditions. "); } else {
    z = rete.executeCommand("(< (fetch RESULT) " + realarea + ")" );
    z2 = z.stringValue(rete.getGlobalContext()); }
```

The Jess codes above have the following interpretation. Firstly, the object *rete* fetches the inputs and asserts them as facts. Secondly the *rete* matches those facts and stores a result based on the inputs. After running the rules, the *rete* fetches the "RESULT" and compares it with the real clear floor area of HS provided by the IM object model. Finally, the Jess execution return value of "z" is converted and reported as a checking result.

5. REFERENCE IMPLEMENTATION

The reference implementation of our concepts proposed in this paper is illustrated in this section. A prototype system for automated building code checking has been developed as a result of our implementation efforts. Case studies of code compliance checking using our implemented prototype are also discussed here.

5.1 Development of the Prototype

The OO representation models of both building design and building code have been implemented in a prototype for automated building code checking in Gintic Institute of Manufacturing Technology, Singapore. The prototype is developed in Java programming language on Windows NT platform. It has two versions. One is for a standalone checking system. Another is for a Web-based client/server checking system. Both versions of the reference implementation provide similar functionality for automated building compliance checking, but use different implementation technologies, including IFC data processing, building rule processing, knowledge base design and implementation, Enterprise Java Bean (EJB) programming, and

J2EE application development. Figure 3 shows the architecture of the prototype system (standalone version).

The standalone checking prototype consists of three main modules: IM Interpreter Module, Rules Engine Module, and Building Code Checking Module.

- The IM Interpreter Module processes IM specifications to form Java class definitions. The module reads in building design CAD files in IFC format, then maps them into building design object models in Java.
- The Rules Engine Module generates rule objects and extracts facts from building codes. It also provides facilities for the execution of rule objects and for the supply of facts to the Building Code Checking Module.
- The Building Code Checking Module allows users to select specific agencies and code clauses as the criteria for performing compliance checks. It communicates with the other two modules for actual design data sets and relevant rule sets, and applies these active rules to design data for compliance checking. A checking report will then be generated.

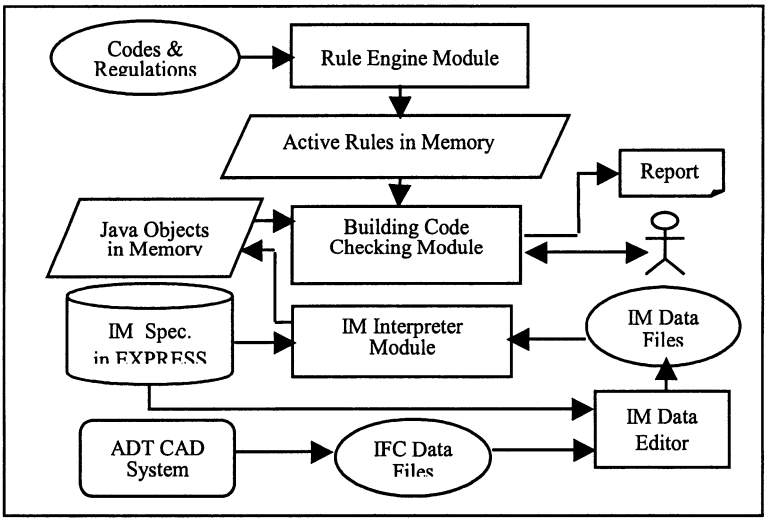


Figure 3. Architecture of the standalone checking prototype

5.2 Case Studies

A CAD design package, Architectural DeskTop (ADT) from Autodesk, is used for the design of a building CAD model (shown in Figure 4) for these case studies of code compliance checking.

Using the ADT package, a standard IFC data file is first created from the CAD model. The file is then processed by an IM data editor to form an IFC-

compliant data file. It is used as the input to the prototype checking system. By using the prototype, over 300 Java classes are generated and installed

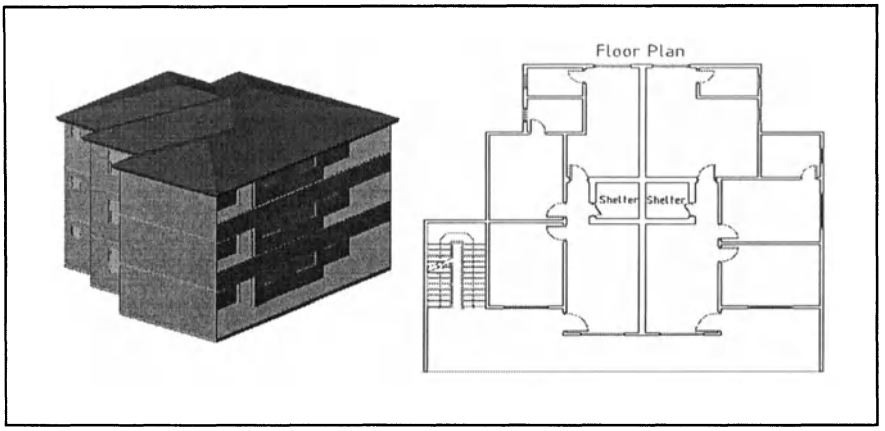


Figure 4. CAD model used for case studies

into an IM conceptual representation model for this sample CAD design. It is populated with the data from the IFC-compliant data file to form an IM object representation model. Any design objects and attributes required for different checking requests related to this building CAD design will be extracted or calculated from this object model.

Suppose the prototype receives such a checking request on the clear area of a household shelter (HS). See Table 2 for the relevant codes used and the Section 3.3 for the knowledge base establishment. As mentioned earlier in the Section 2.3, all the area attribute values are not directly available for this case study. They need to be calculated from other design parameters in the object model. The prototype provides sufficient area calculation algorithms and space location comparison algorithms to support this kind of area calculations. So the area attribute values for the gross floor area and the HS clear area can be generated for this compliance check. On the other hand, an active rule representation model, just for this checking request, is also created in the prototype by using the relevant facts and preconditions stored in the knowledge base. The derived area attributes are then compared with the attributes of rule objects in the active rule model. As such, a checking conclusion will be drawn and reported to the requester. If another checking request comes in, the prototype may create a different active rule representation model to serve this second request. Because the active rule model is quite small, the execution efficiency of the rule objects in the model is satisfactory.

For some checking requests, instead of creating an active rule model for them, the prototype only provides a checking pattern for geometric reasoning over the design parameters. For example, the Reference (PWD, 1997) has such a regulatory requirement on an HS Tower:" HS on every storey are located one on top of the other". In this situation, there are neither facts nor preconditions being extracted from the code. Consequently, no any active rule models can be generated. In order to perform such kind of checking requests, the prototype defines certain checking patterns applying to the IM object model only. The design parameters in the model will be compared to each other using the patterns before a checking result is derived.

Other case studies are also conducted by using the prototype system, including the compliance checks over the dimensions, locations, and relative position relationships of the design object against the established building rules. The prototype provides consistent and reasonable checking reports for all of these testing use scenarios.

6. CONCLUSIONS

This paper presents our concepts and approaches for the representation and execution of building codes in an automated compliance checking process. Object oriented technology is powerful for the establishment of representation models for both building design and building code. IFC-compliant information models, with the OO features, are better representations for building CAD designs in automatic code checking. For building code representation, how to develop rule object models in correspondence with the design object models is most important. In the automated process, the execution of rule objects depends upon not only the rule inference mechanisms provided by a rule engine, but also the geometric reasoning algorithms over the design object model data. The prototype, as a reference implementation of our approaches has been developed. The capabilities of the prototype are demonstrated by case studies. Use of such an IFC-compliant building code checking system would greatly boost the productivity and the standardisation level of information modelling in the building and construction industry.

Further efforts will be made for enhancement of the prototype and in rule objects modelling by IFC methodology.

7. ACKNOWLEDGEMENTS

The authors would like to thank all other team members for their fruitful contributions to the research.

8. REFERENCES

Fenves, S.J., J. H. Garrett, H. Kiliccote, K. H. Law, and K. A. Reed., 1995, "Computer representation of design standards and building codes", *The Int. Journal of Construction Information Technology*, 3(1), p. 13-34.

Kiliccote, H., 1997, "A standards processing framework", *PhD thesis*, Carnegie Mellon University, USA, URL: http://han.ices.cmu.edu

Hakim, M, J.H. Garetta, 1993, "Using description logic for representing engineering design standards", *Journal of Engineering with Computers*, 9, p. 108-124.

Carpenter, B., 1992, "*The logic of typed feature structures with applications to unification grammars, logic programs and constraint resolution*", Cambridge Tracts in Theoretical Computer Science, Cambridge University Press.

Chang, T.W., 1999, "Geometric typed feature structures: towards design space exploration", *PhD thesis*, The University of Adelaide, Australia.

Woodbury, R., A. Burrow, R. Drogenuller, and S. Datta., 2000, "Code checking by representation comparison", Proceedings of *CAADRIA2000*, Singapore, p. 235-244.

Han, C. S., J.C. Kunz, K. H. Law, 1998, "A hybrid prescriptive-/performance-based approach to automated building code checking". Fifth Congress in Computing in Civil Engineering. Boston, MA, USA.

Zarli, A., P. Debras, "Integration of CORBA and WEB technologies in the VEGA DIS", Proceedings of the European Conference on Integration in Manufacturing 1998, Gothenburg, Sweden, p. 184-197.

IAI, 2000, Industry Foundation Classes, Release 2x; http://iaiweb.ibi.gov/

ISO, 1994, "*Industrial Automation Systems and Integration – Product Data Representation and Exchange – Part 11*", Geneva, Switzerland.

Sharpe, R., S. Oakes, 1995, "Advanced IT Processing of Australian Standards and Regulations", *The Int. Journal of construction information technology*, 3(1), p. 73-89.

CORENET, 2000, "*BP-Expert: the intelligent plans checking system*", URL: http://www.corenet.gov.sg/Corenet/bp_expert.html, Singapore.

PWD, 1997, "*Technical Requirements for Household and Storey Shelters*", Lenco Printing Co Pte Ltd, Singapore.

Ernest J.F., 2000, "*Jess, The Java Expert System Shell*", Sandia National Laboratories, Livermore, URL: http://herzberg.ca.sandia.gov/jess.

On top-down architectural lighting design

Constraint-based generation of light sources

Martin Moeck
Department of Architectural Engineering, University of Kansas, USA

Key words: Top-down design, constraint satisfaction optimisation, lighting design, visual performance criteria

Abstract: One key problem of architectural lighting design is to specify goals that relate to aesthetics. Since visibility is an important criterion for many visual tasks and objects, heuristics from industrial lighting and visual inspection can be used to describe the appearance of objects relevant to architectural lighting design, and to derive corresponding light sources. This has the potential to bring computation time in the range of near-interactive rates. A combination of two constraining inputs, which are the specification of desired material appearance and the selection of highlights and shadows can be successfully used in determining light sources.

1. OVERVIEW

1.1 Computer aided lighting design tools

Can the art of architectural lighting be automated? Several top-down tools exist.

Poulin and Fournier present a system where the user can control the definition and position of a light source by manipulating highlights and/or shadows. They use simple geometric constraints and a simple Phong-type illumination model to define the size of a highlight (Poulin, Fournier, 1992).

Schoeneman et al. use linear optimization (Schoeneman, Dorsey, et al,, 1993): the program user "paints" colored light onto surfaces and modifies given initial light positions and intensity distributions, while the computer determines the closest fitting combination of the given lights at user-defined locations.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 331–348.
© 2001 *Kluwer Academic Publishers.*

Kawai et al. use unconstrained optimization techniques (Kawai, Painter, et al., 1993): the lighting designer specifies high-level goals such as "visual clarity" that relate to lighting patterns in diffuse environments. These objective functions are constrained through minimum lighting levels in specific locations, while the computer searches for the "best" possible settings for light source emissivities, surface reflectivities and spotlight directionality". In order to reduce the number of free variables, the program user must select active variables and impose constraints on them.

An approach based on randomly generated, simplified lights produces hundreds of lighting solutions for a simple scene (Marks, Andalman, et al., 1997). The solution generation and its computation time can become prohibitively expensive.

Computer based lighting design techniques based on expert rules change luminaires with respect to architectural elements (Moeck, 1999). Heuristics and expert rules are used to determine the location and rotation of realistic luminaires, given desired light patterns such as a "wash" of light. Scene geometry and material parameters can be of any complexity.

Poulin and Fournier describe another system (Poulin, Fournier, 1995). The user selects colors and applies them to any visible point in the scene. The system attempts to optimize certain functions or to fit values so the color points will remain as close as possible to their assigned colors when the full rendering is completed. The system uses simple illumination models, simple material descriptions, and simplified light sources.

Costa, Sousa, and Ferrera use optimisation to find lighting solutions from the scene geometry, the scene materials and user-specified fictitious luminaires used as design goals (Costa, Sousa, et al., 1999). The desired and undesired locations and emission directions of those luminaires must be specified using a script language.

1.2 A new approach

Most approaches suffer from the fact that lighting expert knowledge is not used. This leads to the selection of constraints that are often not relevant in practice, and to the subsequent generation of trivial solutions. In addition, most approaches do not design the desired dark areas of the scene (Waldram, 1978). Max Reinhardt said, "I am told that the art of lighting consists of putting the light where you want it and taking it away where you don't want it" (Jones, 1941).

This paper presents heuristics-based constraint satisfaction optimization for the top-down lighting design problem: the user describes the desired appearance of selected objects or architectural features in terms of highlights and shadows. Object geometry, material reflectance functions,

and luminaire intensity distributions can be of arbitrary complexity. The design problem is further constrained by specifying the desired appearance of the highlights and the shadows, i.e., strong texture, weak gloss, or flat appearance, etc. Heuristics are used to determine the optimum direction of light. Based on the desired contrast and specific material characteristics, light source locations can be computed (and excluded) such that the desired highlights and shadows are achieved.

2. CONSTRAINT BASED REASONING

Many problems, including planning, can be formulated as Constraint Satisfaction Problems (CSPs) (Shapiro, 1987) (Leler, 1988) (Meseguer, 1989) (du Verdier, Tsang, 1991) (Kumar, 1992) (Guesgen, Hertzberg, 1992) (Kautz, Selman, 1992) (Mackworth, 1992) (Yang, 1992) (Tsang, 1993) (Jampel, Freuder, et al., 1996). A finite CSP is a problem composed of a finite set of variables, each of which is associated with a finite domain, and a set of constraints that restricts the values the variables can simultaneously take. The task is to assign a value to each variable satisfying all the constraints.

2.1 Constraint satisfaction in illumination engineering

In illumination engineering, visibility is a constraint which is used to evaluate luminance distributions that aid the perception of visual signals (Rea, 1993a) (Rea, 1993b) (Rea, 1993c) (SLG, 1992a) (Rea, 1993d). Light source properties to maximize visibility and contrast and minimize veiling reflections have been studied in industrial and office lighting (Rea, 1993e) (Reitmaier, 1979) (Rea, 1993f) (Worthey, 1989a) (Worthey, 1989b) (Worthey, 1990). Real-world applications include visual inspection for quality control of surfaces in manufacturing. Inspection techniques depend on the installation of lighting systems that will maximize the visibility of particular material properties (Rea, 1993e) (IES, 1952) (ANSI, 1972) (SLG, 1992b) (SLG, 1992c), such as scratches and cracks in matte glass, plywood, or stone plates, as well as bumps and dents in metal and plastic sheets, and scratches, granularity, or engravings on polished plates and sheets. Other visual signals include scales and meters behind glass covers, or printed circuit boards on light backgrounds (Coderre, 2000) (Herman, Radice, et al., 2000).

Different lighting systems lend themselves to reveal detail. Important light locations include overhead lighting, grazing light, lighting from the mirror angle, and lighting with the line of sight. By relating light source

properties and light location or direction to specific material properties, it is possible to maximize or minimize the luminance of specific areas. This process can be seen as the CSP.

2.2 Constraints in architectural lighting design

Characteristic features and objects of an architectural setting should have visual emphasis. By relating visual emphasis to luminance contrast, a computer aided lighting design procedure would entail the selection of architectural features as target and background corresponding to highlights and shadows, and the establishment of appropriate luminance differences to enhance their visibility (Moeck, 2000).

The key idea is to specify the spatial relationship between the luminaire and the illuminated object with respect to its surface normal and the viewing direction. Maximum and minimum luminance of various materials depend on specific altitude and azimuth angles between the incident light ray, the surface normal of the illuminated object, the view ray, and the reflective properties of the object (Moeck, 2000) (Worthey, 1989a) (Worthey, 1989b) (Worthey, 1990). These heuristics can be used to optimize the constraints to be applied:

"... heuristics for selecting the right constraints and the right values for these constraints are means for further improvement" (Guesgen, Hertzberg, 1992, p. 65).

Thus, a CSP in lighting design should not only be satisfiable (luminaire locations are found), but optimal, where optimality is defined in application-specific functions according to the domain knowledge (Tsang, 1993, p. 10, p. 300-319). These problems are called Constraint Satisfaction Optimization Problems (CSOP) to distinguish them from the standard CSP. They are successful in CSPs that are too large to be solved by complete algorithms (Jampel, Freuder, et al., 1996, p. 207-216), and they are used here.

2.2.1 Constraint formulation

A CSOP is defined as a CSP together with an optimization function f which maps every solution to a numeric value: (Z, D, C, f), where Z = the finite set of light source location variables defined by their altitude "alt" and azimuth angles "az" with respect to the illuminated object {alt, az};

D = a function which maps every variable in Z to a set of luminaires:

D: Z $\rightarrow$finite set of luminaries.

D_{alt} contains possible values of altitude angles alt. The set D_{alt} is called the domain of alt. The same is true for azimuth angles az.

C = a finite (possible empty) set of constraints on an arbitrary subset of variables in Z.

If S is the set of solutions of (Z, D, C), then f:S → numerical value

The task in a CSOP is to find the solution set with the optimal (maximal or minimal) f-value with regard to the application-dependent optimization function f. Two important methods for tackling CSOPs are the branch and bound algorithms (Lawler, Wood, 1966) (Hall, 1971) (Reingold, Nievergelt, et al., 1977) (Aho, Hopcroft, el al., 1983), and genetic algorithms (Holland, 1975) (Goldberg, 1989) (Davis, 1991) (Tsang, Warwick, 1990). Branch and bound makes use of knowledge of the f function and relies on the availability of good heuristics for estimating the 'best' values. A similar approach will be used here.

2.2.2 Definition of variables

The maximum and minimum luminance of a point on an opaque object depends on the altitude and azimuth of the incident light. Angular values for various material properties (rough-smooth, matte-glossy, textured-flat, light-dark) can be found in the literature (Moeck, 2000) (Reitmaier, 1979) (Worthey, 1989a) (Worthey, 1989b) (Worthey, 1990) (IES, 1952) (ANSI, 1972) (SLG, 1992b) (SLG, 1992c) (Rea, 1993g) (Coderre, 2000) (Herman, Radice, et al., 2000). These values allow the definition of optimization functions, leading to a CSOP. In order to express the desired appearance of highlights and shadows of an object to be illuminated as constraints, the interface shown in figure 1 is used.

Figure 1. Appearance parameters used as constraints

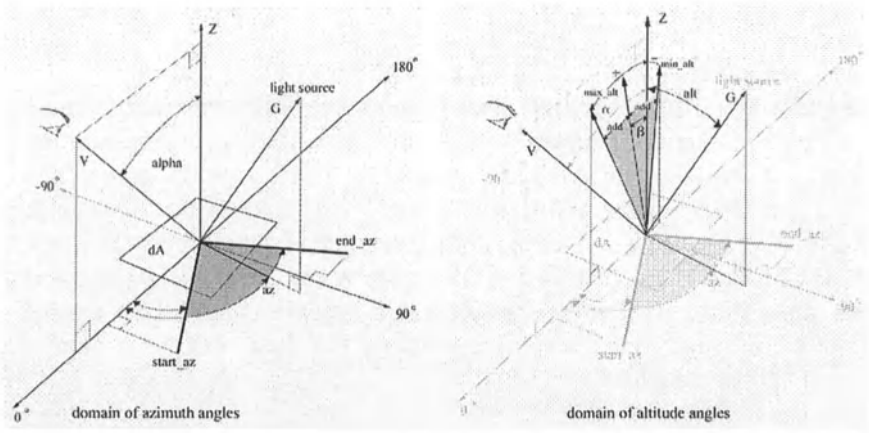


Figure 2. The direction G from the highlight to the light source is to be determined. The possible range of azimuth angles of G starts at start_az and ends at end_az, while the range of possible altitude angles for G starts at min_alt and ends at max_alt.

Figure 2 shows a point on an object to be illuminated, given a view point. Its luminance is to be maximized or minimized. Its surface normal is Z, and the eye ray is V. The direction vector to the light source is G. G is defined by its altitude and azimuth angles az and alt with respect to to V and Z. alt and az need to be found for a set of points on an architectural feature or object with specific material properties.

3. IMPLEMENTATION

3.1 Optimisation functions

The material properties corresponding to the constraints of figure 1 depend on the light source location, and should be enhanced or suppressed. For example, the scale variable gloss operates from "no gloss" (gloss = 0) to "glossy" (gloss = 100). A desired glossy appearance means to place luminaires at the mirror angle, no gloss means to place the light along the eye ray direction. A desired strong texture places lights at the grazing angle. Texture suppression is achieved by placing lights along the surface normal of the object.

The optimization functions shown in tables 1 - 3 determine the light source direction G for highlights and shadows. These constrain the values of Z{alt, az} using heuristics. Input is from the four scale values operating from 0 (left value) to 100 (right value) from figure 1.

Table 1. "Appearance of a glossy surface"
Scalevalue = gloss; alpha = 45 degrees
No gloss: gloss = 0; glossy: gloss = 100

Formula for angle	Angle values for gloss = 0	Angle values for gloss = 25	Angle values for gloss = 50	Angle values for gloss = 75	Angle values for gloss = 100
If gloss <= 50 then start_az = 0 else start_az = 180 – 2*((abs(gloss-100)/100)*90)	0	0	0	135	180

If gloss <= 50 then
end_az = start_az + (gloss/50)*90
else
end_az = 180
	0	45	90	180	180

add = 50 – abs(gloss –5 0)*(90/50)	0	45	90	45	0
beta = (-(gloss-50))*alpha/50	alpha	22.5	0	-22.5	-alpha
max_alt = beta + add	alpha	67.5	90	22.5	-alpha
min_alt = beta - add	alpha	-22.5	-90	-67.5	-alpha

Table 2. "Appearance of a textured surface"
Scale variable: texture; alpha = 45 degrees
Weak texture: texture = 0; strong texture: texture = 100

Formula for angle					
start_az = 0	0	0	0	0	0
end_az = 180	180	180	180	180	180
add = (50 – abs(texture – 50))*90/50	0	45	90	45	0
If texture < 50 beta = 0 else beta1 = (texture-50)*90/50 beta2 = -beta1	beta = 0	beta = 0	beta1 = 0 beta2 = 0	beta1 = 45 beta2 = -45	beta1 = 90 beta2 = -90
If texture < 50 max_alt1 = beta + add min_alt1 = beta – add else max_alt1 = beta1+ add min_alt1 = beta1 – add max_alt2 = beta2+ add min_alt2 = beta2 - add	max_alt1 = 0 min_alt1 = 0	max_alt1 = 45 min_alt1 = -45	max_alt1 = 90 min_alt1 = -90 max_alt2 = 90 min_alt2 = -90	max_alt1 = 90 min_alt1 = 0 max_alt2 = 0 min_alt2 = -90	max_alt1 = 90 min_alt1 = 90 max_alt2 = -90 min_alt2 = -90

Table3. "Appearance of a bumpy surface"
Scale variable: eyeray; alpha = 45 degrees
Flat appearance: eyeray = 0; bumpy appearance: eyeray = 100

Formula for angle	Angle value for eyeray = 0	Angle value for eyeray = 25	Angle value for eyeray = 50	Angle value for eyeray = 75	Angle value for eyeray = 100
start_az = 0	0	0	0	0	0
If eyeray < 50	20	100	180	180	180
end_az = 3.2*eyeray + 20					
else					
end_az = 180					
add=(50–abs(eyeray–50)) * 90/50	0	45	90	45	0
If eyeray < 50	beta = 45	beta =	beta1 = 0	beta1 =	beta1 =
beta=-1*(eyeray-50)*		22.5	beta2 = 0	45	90
alpha/50				beta2 =	beta2 =
else				-45	-90
beta1 = (eyeray-50)*90/50					
beta2 = -beta1					
If eyeray < 50	max_alt1	max_alt1	max_alt1	max_alt1	max_alt1
max_alt1 = beta + add	= 45	= 67.5	= 90	= 90	= 90
min_alt1 = beta – add	min_alt1	min_alt1	min_alt1	min_alt1	min_alt1
else	= 45	= -22.5	= -90	= 0	= 90
max_alt1 = beta1+ add			max_alt2	max_alt2	max_alt2
min_alt1 = beta1 – add			= 90	= 0	= -90
max_alt2 = beta2+ add			min_alt2	min_alt2	min_alt2
min_alt2 = beta2 - add			= -90	= -90	= -90

The scale "Appearance of matte detail and layers" is very similar to the scale "Appearance of a textured surface" and its functions are omitted here.

The final solution is the angular range with the largest minimum and the smallest maximum of the four domains. The following constraints serve as an example. Desired is a highlight with an appearance described in figure 1. The following scale values are desired:

"Appearance of a glossy surface": gloss = 85
"Appearance of a textured surface": texture = 75
"Appearance of matte detail and layers": illum = 25
"Appearance of a bumpy surface": eyeray = 60

The angle alpha between the light of sight and the surface normal of the highlight is 45 degrees. Based on tables 1-3, the angular ranges for alt and az are as follows:

altitude angles for gloss: from -58.5 to -4.5 degres
azimuth range for gloss: 153.0 to 180 degrees
altitude angles for texture: from 0.0 to 90.0 and from -90.0 to 0.0 degrees
azimuth range for texture from 0 to 180 degrees
altitude angles for illum: from 0.0 to 90.0 and from -90.0 to 0.0 degrees
azimuth range for illum from 0 to 180 degrees
altitude angles for eyeray: from -54.0 to 90.0 and from -90.0 to 54.0 degrees
azimuth range for eyeray from 0 to 180 degrees

Therefore, the final altitude range is from -4.5 to -58.5 degrees, while the azimuth range is from -153 to -180 and from 153 to 180 degrees.

Figure 3 shows the corresponding light source arrangements with respect to a small square patch selected as the "highlight". The arrows correspond to vectors Z, x, and y from figures 2 and 3.

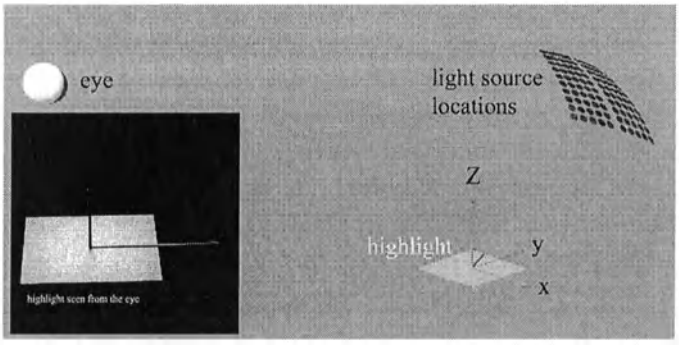


Figure 3. Highlight coordinate system and luminaire locations based on the constraints of figure 1

The implementation of the illumination of a 3D object based on constraints of figure 1 follows. Desired are the highlights and shadows on an object shown in figure 4.

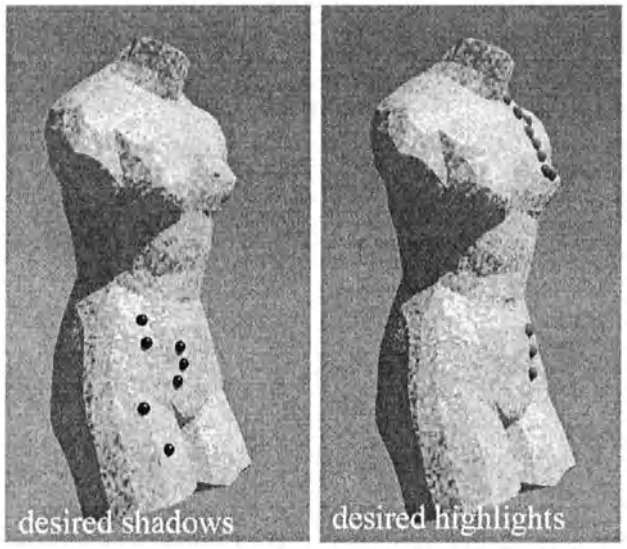


Figure 4. Desired highlights and shadows on a glossy, textured object

3.2 Algorithms

The resulting spotlights are arrived at as follows in algorithm 1:

3.2.1 Algorithm 1:

input: start_az, end_az, min_alt, max_alt

```
for each highlight
   for each view point
      determine vectors V, Z, x, and y
      az = start_az
      alt = min_alt
      while az <= end_az
         while alt <= max_alt
            calculate vector G with altitude angle alt and azimuth angle az
            create a light source with surface normal -G
            starting at the highlight, fire G into the scene
            determine the intersection with the nearest scene element: S
            place the light source with surface normal –G at S
            alt = alt + increment
         end while
         az = az + increment
```

```
      end while
   end for
end for
```

See figure5 for the object in an interior setting.

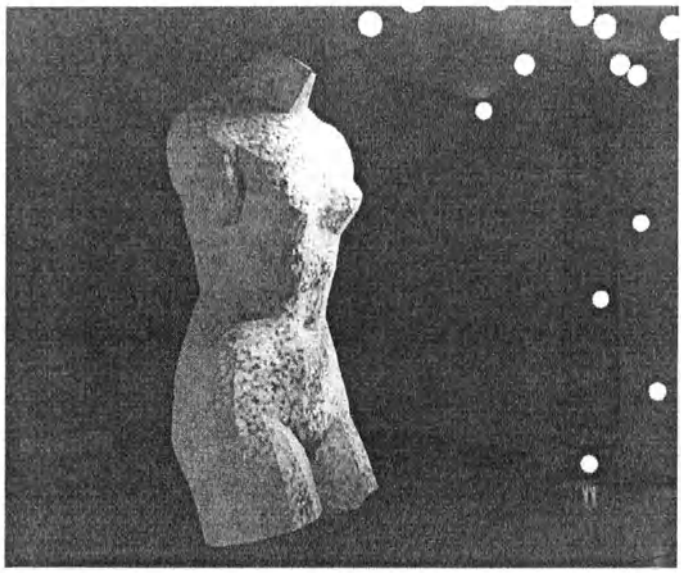


Figure 5. Object with illuminated highlights. The desired dark areas are partially illuminated.

The angular spacing between the luminaires (variable "increment" of algorithm 1) is 5 degrees. The 48 generated light sources are commercially available low voltage incandescent spotlight with a parabolic reflector and a very tight cutoff angle. Nevertheless, the desired dark areas are illuminated. This is due to the non-parallel emission characteristics of the spots. Therefore, it is necessary to introduce an algorithm to locate the spots interfering with the desired dark areas, as shown in algorithm 2. These spots need to be eliminated, reducing the domain D{az, alt}.

3.2.2 Algorithm 2

```
for each dark area
  for each view point
     determine vectors V, Z, x, and y
     az = start_az
     alt = min_alt
     while az <= end_az
```

```
        while alt <= max_alt
            calculate vector G with altitude angle alt and azimuth angle az
            starting at the dark area, fire G into the scene
            determine the intersection with the nearest scene element: A
            alt = alt + increment
        end while
        az = az + increment
    end while
end for
```

See figure 6 for luminaire locations and absorbers with locations "A". It is obvious that some spots interfere with the desired dark areas.

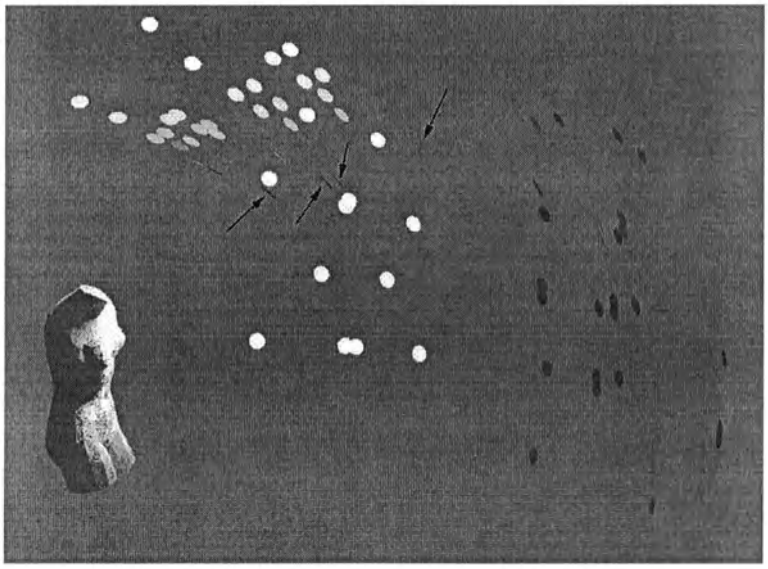


Figure 6. Some of the 48 luminaires with 5 degree angular spacing interfere with absorbers marked with arrows and must be removed.

The domain of possible light locations from algorithm 1 is reduced by eliminating all spots which are closer than a certain distance to any absorber location in A. For this particular case, it is approximately 2 m. The resulting illuminated object with the desired shadows and its reduced domain of 12 spotlights is shown in figure 7.

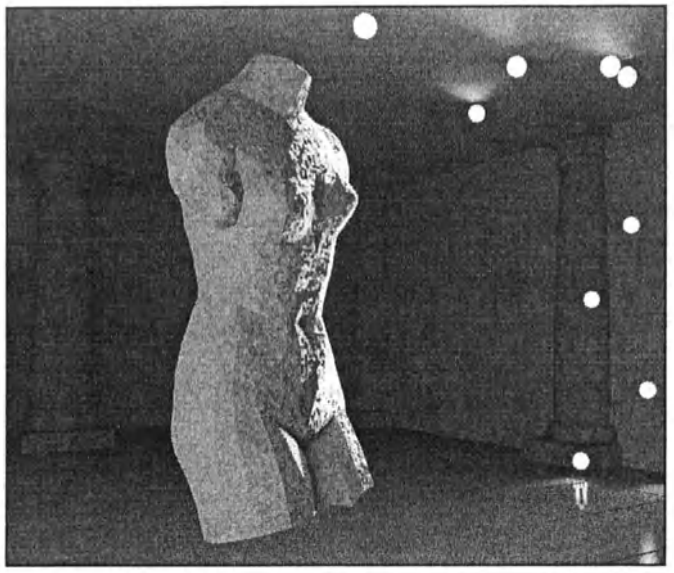


Figure 7. Illuminated object with 36 interfering spots removed. The remaining 12 lights fulfil three constraints: the highlights reflect light in the eye (high gloss), texture is emphasized through a grazing angle of light incidence, and the desired shadow areas are not illuminated.

Although the highlights and dark areas rather close, the desired luminance pattern is achieved, while the remaining population is still large enough to select individual spots. This would not have been possible using educated expert guesses about light source locations, because the 3D geometry of the illuminated object is too complex.

4. PROGRAM IMPLEMENTATION

This program uses the Radiance lighting simulation system for rendering (Ward, 1994). The code for the tool discussed is implemented in Tcl/Tk under Unix (Osterhout, 1994), and consists of the following stand-alone programs:

- a material selection tool. It allows the definition and assignment of a wide range of materials to the object to be illuminated.
- a viewer tool. It allows the definition and manipulation of viewpoints, view directions, view types, angular view sizes, zoom functions, view rotations, and so forth.
- the constraint definition tool shown in figure 1, which imposes constraints on the appearance of highlights and shadows

- a luminaire editing tool to change light source characteristics (wattage, size, cutoff angle, luminaire selection)
- an application for the selection of highlights and shadows on the object to be illuminated. This program manages the interaction of the other programs.

The stand-alone programs communicate via the send command. With send, any Tk-based application can invoke arbitrary Tcl scripts in any other Tk application on the display, allowing the development of reusable, small software packets.

5. DISCUSSION

In principle, highlights can be illuminated from 64080 possible light sources, assuming 1 degree angular spacing between the luminaires. This domain is too large to do exhaustive searches. However, different material characteristics (bumpy or flat, glossy or matte, textured or smooth) will yield drastically different appearances of highlights. These appearances relate to the amount of visible detail in the illuminated surface and can be used as design intentions to constrain the desired look of the highlighted material. Heuristics from industrial lighting and visual inspection techniques are used to derive the corresponding light incidence angles. These expert rules limit the angles to useful values. For example, a desired glossy appearance requires to place a light source at altitude angle -alpha and azimuth angle 180 degrees (see figure 2), while a surface without a glossy look requires an azimuth angle of 0 degrees and an altitude angle of alpha.

The formulae from tables 1 - 3 serve as a starting point for maximizing the luminance of highlights, and for minimizing the luminance of shadows. In the future, more comprehensive formulae should be developed to derive the light incidence angles. These should consider more sophisticated material parameters and more constraints or appearance descriptors, and could even consider mixed material properties (i.e. black ink on glossy paper). On the other hand, appearance descriptors need to be simple and intuitive to be understood by the lay program user.

Lighting design based on material properties requires rendering systems with material definitions based on bi-directional reflectance functions. The tool should be extended to include a building material library of known reflectance functions. In this way, the light incidence angles at which maximum luminance occurs could be calculated, in addition to using heuristics, and results could be fine tuned.

It has been claimed that lighting designers are also shadow designers, because they keep light away from desired dark areas. This is problematic

due to spill light. Light beams are not parallel and will illuminate highlights and desired shadows. This problem has been solved by introducing an algorithm which locates light sources which conflict with the shadow areas. The initial luminaire population is screened and detrimental lights are eliminated, leaving a useful selection as a starting point for the final manual lighting design implementation. In the future, additional constraints could be easily imposed by selecting luminares based on their location (wall, ceiling, floor), orientation, distance, etc. In addition, the effect of a range of different view points and viewing directions needs to be tested.

The computation time is around 8 minutes for the case shown (400 Mhz processor with 128 MB RAM). This time is mainly a function of the desired angular spacing of the luminaires (see angle increment in algorithms 1 and 2). It also depends on how tight the constraints are set and how large the resulting solution space is. The expert rules expressed as formulae in tables 1 - 3 speed up generation significantly.

6. OUTLOOK AND FUTURE APPLICATION TO COMPLEX ARCHITECTURAL SETTINGS

This paper introduced the theory of CSOPs to top-down lighting design, using an example with one viewpoint, and one illuminated object with eleven highlights and seven shadow points. Future studies will determine the suitability of the algorithms to more complex architectural settings. This will include furnished rooms with various objects on display, many different viewpoints, and multiple constraints for various highlights and shadows scattered widely across the room. Additional constraints will exclude direct glare at the eye positions of observers exploring the space from various vantage points. This will make the study quite complex. It will determine the effects of multiple constraints, how their settings affect the size of the solution space, and how multiple viewpoints and additional glare constraints affect previous solutions. This is beyond the scope of this paper.

7. CONCLUSION

The advantages of constraint-based optimization are the setup of meaningful constraints, and the reduction of a very large solution space to a small domain. This makes computer aided lighting design a tractable problem. One key problem of CAAD is to relate constraints to esthetics, and the setup of lighting design goals which allow to specify the desired design product. Since visibility is an important criterion for many visual tasks and

objects, heuristics from industrial lighting and visual inspection can be used to describe the appearance of objects relevant to architectural lighting design, and to derive corresponding light sources. This has the potential to bring computation time in the range of near-interactive rates. It has been shown that a combination of two constraining inputs, which are the specification of desired material appearance and the selection of highlights and shadows, or dark areas, can be a successful lighting design goal for determining light sources.

8. REFERENCES

Aho, A.V., J.E. Hopcroft, and J.D. Ullman, 1983, *Data structures and algorithms*, Addison-Wesley.

ANSI PH2.32-1972, 1972, *American national standard viewing conditions for the appraisal of color quality and color uniformity in the graphic arts*, American National Standards Institute, New York.

Coderre, J., 2000, "Machine Vision Illuminates the Future of Electronics", *Photonics Spectra*, December 2000, p. 90-92.

Costa, A.C., A.A. Sousa, and F. N. Ferreira, 1999, "Lighting Design: A Goal Based Approach Using Optimisation", *Rendering Techniques '99, Proceedings Eurographics Workshop*, Springer Verlag, Wien, p. 317-328.

Davis, L. (ed.), 1991, *Handbook of genetic algorithms*, Van Nostrand Reinhold, New York.

du Verdier, F, and J.-P. Tsang, 1991, "A spatial reasoning method by constraint propagation", *Proceedings 11th International Workshop of Expert Systems and Their Applications*, p. 297-314.

Goldberg, D.E., 1989, *Genetic algorithms in search, optimization, and machine learning*, Addison-Wesley.

Guesgen, H.W., and J. Hertzberg, 1992, *A Perspective of Constraint-Based Reasoning*, Lecture Notes in Computer Science, Springer-Verlag, Berlin.

Hall, P.A.V. 1971, "Branch and bound and beyond", *Proceedings International Joint Conference on AI*, pp. 641-650.

Herman, J.W, J. Radice, J. Disciullo, and P. Chipman, 2000, "Flexible lighting modules", *Advanced Packaging*, May 2000, p. 27-32.

Holland, J.H., 1975, "Adaptation in natural and artificial systems", University of Michigan Press.

IES Committee on Lighting Study Projects in Industry, Subcommittee on Supplementary Lighting, 1952, "Recommended practice for supplementary lighting", *Journal of the Illuminating Engineering Society* 47(11), p. 623-635.

Jampel, M., E. Freuder, and M. Maher, 1996, *Over-Constrained Systems*, Lecture Notes in Computer Science, Springer-Verlag, Berlin.

Jones, R. E., 1941, *The Dramatic Imagination*, Duell Sloan & Pearce, New York, p. 111.

Kautz, H., and B. Selman, 1992, "Planning as satisfiability", *Proceedings 10th European Conference on Artificial Intelligence*, 1992, p. 360-363.

Kawai, J.K., J.S. Painter, and M.F. Cohen, 1993, "Radioptimization - Goal Based Rendering", *Computer Graphics Proceedings, SIGGRAPH 1993*, p. 147-154.

Kumar, V., 1992, "Algorithms for constraint-satisfaction problems: a survey", *AI Magazine*, Vol. 13, No. 1, Spring 1992, p. 32-44.

Lawler, E.W., and D.E. Wood, 1966, "Branch-and-bound methods: a survey", *Operations Research*, Vol. 14, 1966, p. 699-719.

Leler, W., 1988, *Constraint programming languages, their specification and generation*, Addison-Wesley, 1988.

Mackworth, A.K., 1992, "The logic of constraint satisfaction", *Artificial Intelligence*, Vol. 58, Nos. 1-3, p. 3-20.

Marks, J., B. Andalman, P.A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, and T. Kang, 1997, "Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation", *Computer Graphics Proceedings, SIGGRAPH 1997*, pages 389-400.

Meseguer, P., 1989, "Constraint satisfaction problems: an overview", *AI Communications*, Vol. 2, No. 1, March 1989, p. 3-17.

Moeck, M. 2000, "Lighting design based on luminance contrast", *Lighting Research and Technology* 32(2), 2000, p. 55-64.

Moeck, M., 1999, "On Computer Aided Lighting Design: Lighting Techniques", *Journal of the Illuminating Engineering Society of North America* 28(2), p.33-41.

Osterhout, J., 1994, *Tcl and the Tk Toolkit*, Addison-Wesley, Reading, Massachusetts.

Poulin, P., and A. Fournier, 1992, "Lights from highlights and shadows", *Computer Graphics*, volume 25, p. 31-38.

Poulin, P., and A. Fournier, 1995, "Painting surface characteristics", Rendering Techniques '95, *Proceedings Eurographics Workshop*, Springer-Verlag, Wien, p. 160-169.

Rea, M. (ed.), 1993a, "Visibility for Specific Visual Tasks", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 664.

Rea, M. (ed.), 1993b, "Visual ability", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 86-89.

Rea, M. (ed.), 1993c, "Visual Range of Luminous Signals", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 93-91.

Rea, M. (ed.), 1993d, "Luminance Contrast", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 83.

Rea, M. (ed.), 1993e, "Industrial Lighting", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 629-684.

Rea, M. (ed.), 1993f, "Office Lighting", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 517-540.

Rea, M. (ed.), 1993g, "Classification of Visual Tasks and Lighting Techniques", *IESNA Lighting Handbook*, 8th edition, IESNA, New York, p. 636-638.

Reingold, E.M., J. Nievergelt, J., and N. Deo, 1977, *Combinatorial algorithms: theory and practice*, Prentice Hall.

Reitmaier, R., 1979, "Some effects of veiling reflections in papers", *Lighting Research and Technology* 11(4), p. 204.

Schoeneman, C., J. Dorsey, B. Smits, J. Arvo, and D. Greenberg, 1993, "Painting with Light", *Computer Graphics Proceedings, SIGGRAPH 1993*, p. 143-146.

Shapiro, S.C., 1987, *Encyclopedia of Artificial Intelligence*, John Wiley & Sons, Chichester, England.

SLG Schweizerische Lichttechnische Gesellschaft (ed.), 1992a, "I-2 1.2.2 Hellempfindlichkeit", *Handbuch für Beleuchtung*, Ecomed Verlag, Landsberg, Germany, p. 2-3.

SLG Schweizerische Lichttechnische Gesellschaft (ed.), 1992b, "Tab. II-2.1.4: Sonderaufgaben in der Industrie", *Handbuch für Beleuchtung*, Ecomed Verlag, Landsberg, Germany, p. 4-14.

SLG Schweizerische Lichttechnische Gesellschaft (ed.), 1992c, "Tab. II-2.1/2: Örtliche Sehaufgabe am Arbeitsplatz - Sehaufgaben und praktische Lösungen", *Handbuch für Beleuchtung*, Ecomed Verlag, Landsberg, Germany, p. 4-14.

Tsang, E., 1993, *Foundations of Constraint Satisfaction*, Academic Press, London.

Tsang, E.P.K., and T. Warwick, 1990, "Applying genetic algorithms to constraint satisfaction problems", *Proceedings European Conference on AI*, p. 649-654.

Waldram, J.M., 1978, "Designed Appearance Lighting", *Developments in Lighting-1*, Chapter 5, Lynes, J.A., (ed.), Applied Science Publishers, London.

Ward, G.J., 1994, "The Radiance Lighting Simulation and Rendering System", *Computer Graphics Proceedings, SIGGRAPH 1994*, p. 459-472.

Worthey, J. A., 1989a, "Geometry and amplitude of veiling reflections", *Journal of the Illuminating Engineering Society* 18(1), p. 49-62.

Worthey, J. A., 1989b, "Effect of veiling reflections on the vision of colored objects", *Journal of the Illuminating Engineering Society* 18(2), p. 10-15.

Worthey, J. A., 1990, "Lighting quality and light source size", *Journal of the Illuminating Engineering Society* 19(2), p. 142-148.

Yang, Q., 1992, *Artificial Intelligence*, Vol. 58, Nos. 1-3, p. 361-392.

Computationally rendered architectural spaces as means of lighting design evaluation

Hesham Eissa[1], Ardeshir Mahdavi[1], Roberta Klatzky[2], Jane Siegel[3]
[1]*School of Architecture, Carnegie Mellon University*
[2]*Department of Psychology, Carnegie Mellon University*
[3]*Human-Computer Interaction Institute, Carnegie Mellon University*

Key words: Lighting simulation, Computational visualization, Subjective lighting evaluation

Abstract: This paper describes an empirical study that was conducted to determine whether and to what extent subjective evaluation of lighting in architectural spaces can be reproduced using computationally rendered images of such spaces. The results imply that such images can reliably represent certain aspects of the lighting conditions in real spaces.

1. INTRODUCTION

Scientific simulation of light distribution in architectural spaces has been used in the past to generate numeric values of lighting performance indicators such as illuminance levels and daylight factors (e.g. on task surfaces), luminance levels, and glare indices. Designers and consultants typically compare such numeric results with minimum or maximum requirements in relevant illuminating engineering standards to decide if a particular design meets mandated performance criteria. Recently, it has been suggested that such traditional numeric evaluation methods may be complemented (or even substituted) by approaches that rely on scientific visualization tools, enabling the users to virtually observe the illuminated space and "directly" evaluate its lighting. The reasoning is that such scientific visualizations combine photo-realistic rendering with photometric accuracy, thus providing an image of the architectural space that is a dependable representation of its lighting. Consequently, designers could use

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 349–360.
© 2001 *Kluwer Academic Publishers.*

computational visualization of lighting conditions in spaces in order to judge their visual quality. We consider this idea as a hypothesis in need of empirical testing. In this paper, we describe such a test.

2. PRIOR RESEARCH

Prior research investigated the effect of light on impressions of activity setting or mood. Also this research considered how we can recognize spatial lighting patterns as part of visual language that can assist the designer in implementing such impressions. Flynn tested the lighting-cue theory by accumulating data through *semantic differential scaling* and *multidimensional scaling* techniques (Flynn et al. 1973). The results showed how a system of visual cues could be recognized and interpreted in consistent ways by those who share a cultural background. This implied that it is possible to formulate standards for certain subjective aspects of lighting design.

In this context, visualizations that would resemble the real space could facilitate a more efficient and less costly procedure to investigate non-quantitative factors in lighting design. Hendrick et al. 1977 replicated Flynn's study to determine whether the subjective response to the visual experience of a space, as represented by two-dimensional slide projections, is comparable with the subjective response to real spaces. Their results suggest that there may be a potential to use (two-dimensional) slides of architectural spaces as representations of the lighting conditions in actual (three-dimensional) spaces. They also recommended that further research should establish a standardized method of photographing spaces to reproduce the effects of those spaces with little error (Hendrick et al. 1977).

In a more recent study by Veitch and Newsham (1996), the computational visualization tool Radiance was used to simulate different lighting settings in a workplace. The idea was to investigate the lighting designers' evaluations of lighting quality based on computer-generated renderings. The test lighting conditions were defined by a 3x3 matrix of levels of Lighting Power Density (LPD) and lighting quality as defined by designers (Designer' Lighting Quality, DLQ). The test of these lighting quality levels showed a remarkable disagreement in the overall ratings (overall lighting quality was rated on 1 to 9-scale), while semantic differential scale ratings showed better agreement. Veitch and Newsham recommended that future research should systematically investigate the *validity of rendering tools* both for lighting design and as means of representing lighting alternatives to clients. Also that study confirmed the

reliability of *semantic differential scaling* as a subjective measure of a lit environment (Veitch and Newsham, 1996).

3. EXPERIMENTAL STUDY

3.1 Overview

The goal of the experimental analysis was to see if and to what extent the subjective lighting evaluation of computationally rendered images of spaces is consistent with subjective lighting evaluation of real spaces. The analysis involved the following steps:

a) Based on the results of previous research, a metric was selected that captures certain subjective light quality dimensions. This metric was then slightly modified based on the results of a bipolar semantic survey, and an experiment involving test participants evaluating lighting attributes of images of various architectural spaces.
b) Five actual lighting situations were selected involving three spaces (an office, a conference room, and a computer cluster) and different lighting schemes.
c) These situations were evaluated by a first group of test participants using the above-mentioned subjective lighting metric.
d) High-quality renderings of the above-mentioned situations were generated using an advanced visualization tool. The rendered images satisfied both the visual resemblance condition (sufficiently realistic depiction of the real lighting situations) and photometric reliability (sufficiently accurate replication of photometric measurements in the real spaces).
e) The rendered versions of the lighting situations were evaluated by a second group of test participants, using again the subjective lighting metric.
f) Subjective lighting assessments of the real spaces were compared with those of the computational visualizations to empirically determine the degree to which such visualizations can represent real spaces toward subjective lighting evaluation of architectural designs.

Sections 3.2 to 3.7 below describe these steps in more detail.

3.2 A subjective lighting metric

A common method to capture subjective impressions of otherwise non-measurable phenomena is the use of semantic differential rating scales. These include pairs of terms on a bipolar scale that addresses various

dimensions of people's subjective impressions. For the purposes of this study, four steps were undertaken:

a) Suggestions for such bipolar pairs of terms were collected via a survey of 44 Architecture students;
b) The scales resulting from this survey were compared with the semantic differential scales developed by Flynn et al.1973;
c) Taking both the results of the survey and Flynn's scales, 28 pairs of terms were selected for further consideration;
d) This group of terms was tested using a small group of test participants who used it to evaluate the lighting quality of a number of office spaces as projected in slides. The final metric was derived based on a statistical analysis of the results of this test. Principal component analysis (PCA) was used as a data reduction method to eliminate the redundancy among the selected scales. The resulting scales are 10 pairs of terms under 7 categories (cp. Table 1). This differs only slightly from the scale adapted by Flynn et al. 1973. As an example, figure 1 illustrates one of the 7-point bipolar scales used in this study.

Table 1. The 10 selected semantic differential rating scales

Category	Terms
1. Evaluative (Psychological Impression)	Pleasant - Unpleasant
	Interesting - Boring
	Cheerful - Somber
	Shiny - Dull
2. Perceptual Clarity (Brightness)	Bright - Dim
3. Spaciousness	Large - Small
4. Light Distribution	Uniform - Non uniform
5. Spatial Complexity	Simple - Complex
6. Formality	Private - Public
7. Thermal, acoustic, and haptic associations	Cool – Warm

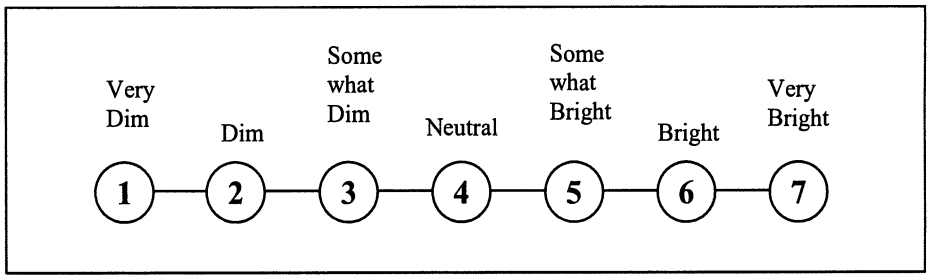


Figure 1. Example of a scale (Dim/Bright) of the derived metric (semantic differential)

3.3 Spaces and scenes

To capture people's subjective impressions of lighting, five actual lighting situations (scenes) were selected. These involved three spaces (an office, a conference room, and a computer cluster) and different lighting schemes (cp. Table 2). Schematic plans of spaces 1-A, 1-B, 2, 3-A, and 3-B are given in figures 2 to 4. Digital photographs of scenes 1-A, 2, and 3-A are given in figures 5 to 7.

Table 2. The five spaces and lighting schemes

Scene	Space	Lighting scheme
1-A	Office space 1	Indirect fluorescent
1-B	Office space 1	Spot – down light
2	Conference room	Indirect Halogen
3-A	Computer cluster (Large)	Direct fluorescent
3-B	Computer cluster (Small)	Direct fluorescent

3.4 Evaluating real spaces

A group of 50 people participated in the subjective evaluation of the above-mentioned scenes. The majority of the participants were senior architecture students, since this study was particularly motivated by the suggested potential of computational visualization as means of lighting design evaluation. The test participants observed the spaces from predefined vantage points (cp. the schematic plans, figure 2 to 4) and evaluated those using the semantic differential scale.

3.5 Renderings

The aforementioned five lighting situations were rendered using a commercially available visualization tool Lightscape release 3.2 (Lightscape 1999). The input information was derived either from direct measurements of the relevant attributes in the real spaces (e.g. reflectance of the walls) or extracted from documentation of the relevant equipment (light sources and luminaries). Figures 5 to 7 include the resulting renderings for scenes 1-A, 2, and 3A along with digital photographs of the corresponding (real) spaces.

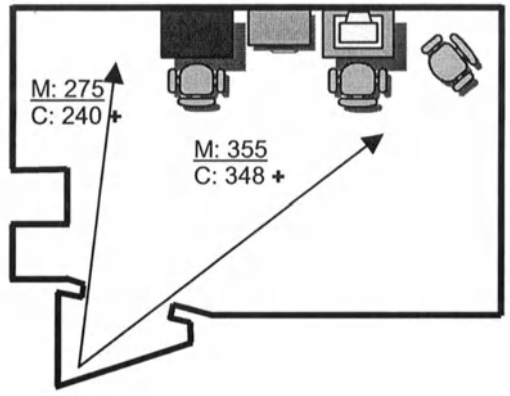


Figure 2. Schematic plan of space 1-A and 1-B (2 settings: fluorescent and down light) with indication of observers' vantage point as well as examples of measured (m) and computed (c) horizontal illuminance levels in lux (for the fluorescent lighting setting 1-A)

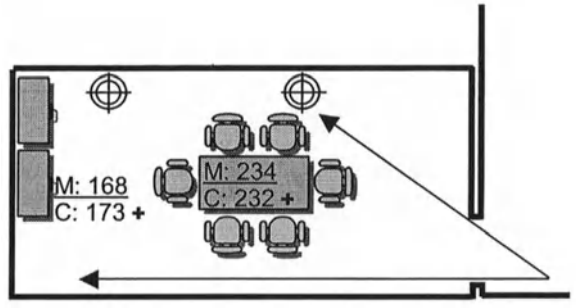


Figure 3. Schematic plan of space 2 with indication of observers' vantage point as well as examples of measured (m) and computed (c) horizontal illuminance levels in lux

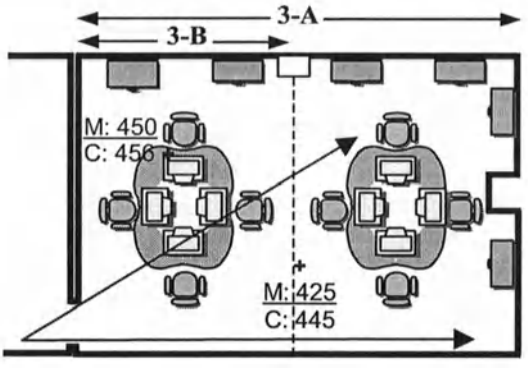


Figure 4. Schematic plan of space 3-A and 3-B with indication of observers' vantage point as well as examples of measured (m) and computed (c) horizontal illuminance levels in lux

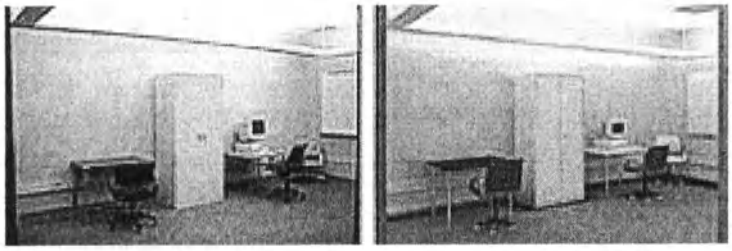


Figure 5. Space 1-A (Left: actual, Right: rendering)

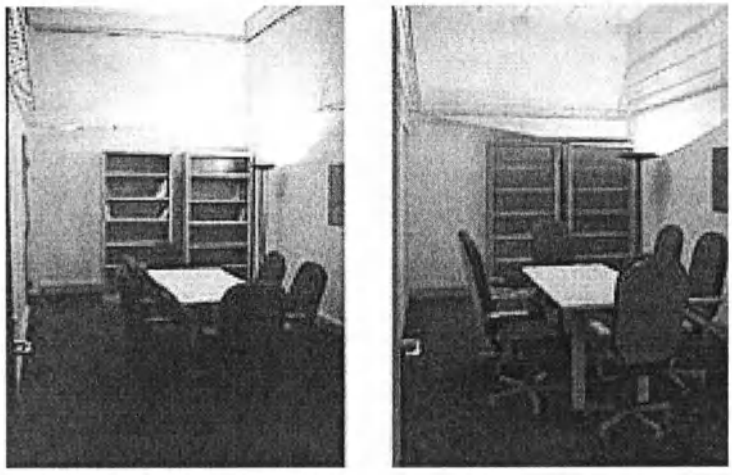


Figure 6. Space 2 (Left: actual, Right: rendering)

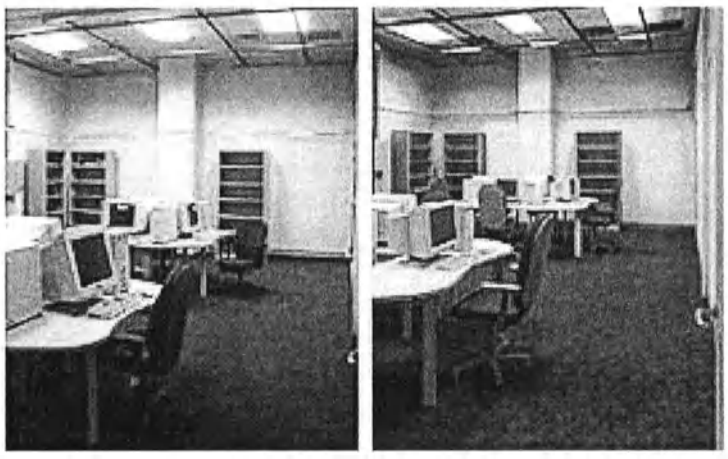


Figure 7. Space 3-A (Left: actual, Right: rendering)

3.6 Evaluating rendered images

A second group of 50 people participated in the subjective evaluation of the five scenes based on their visualizations and used the same semantic differential scale. To avoid carry-over effects, no individual participated in both tests. Moreover, the profile of the second group of the test participants matched that of the first group in terms of age, gender, and educational background. The rendered scenes were shown to the test participants on a 19" colored computer monitor.

3.7 Results

The descriptive statistics for both tests are summarized in table 3. Participants' responses on a scale of 1 to 7 were recorded and the table shows the mean and standard deviation of their subjective impressions of actual spaces and the corresponding renderings. A subset of these results (mean evaluations for scenes 1-A, 1-B, and 3-A as compared to their respective renderings across the 10 scales of the semantic differential) is also illustrated graphically in figures 8 to 10.

Table 3. Descriptive statistics in terms of mean and standard deviation (in parenthesis) of evaluations of both actual scenes (S) and their renderings (R)

Scene:	1-A		1-B		2		3-A		3-B	
	S	R	S	R	S	R	S	R	S	R
1. Dim–Bright	4.6 (1.5)	4.9 (1.3)	2.1 (1)	1.8 (0.8)	4 (1.2)	3.1 (0.8)	5.7 (1.1)	5.6 (1.3)	6.2 (0.8)	5.8 (1)
2. Non uniform–Uniform	5.3 (1.5)	4.9 (1.6)	1.9 (0.9)	2.6 (1.3)	3.1 (1.6)	2.4 (1.3)	5.4 (1.4)	5.2 (1.6)	5.7 (1.3)	5.3 (1.3)
3. Boring–Interesting	3.0 (1.3)	3.4 (1.5)	4.8 (1.2)	4.4 (1.4)	5.3 (1.1)	4.8 (1.2)	4.1 (1.3)	3.5 (1.2)	3.6 (1.5)	3.0 (1.3)
4. Private–Public	5.0 (1.2)	4.6 (1.5)	2.6 (1.1)	2.9 (1.4)	3.7 (1.6)	3.5 (1.1)	6.0 (0.9)	5.7 (1.3)	5.6 (1.2)	5.0 (1.3)
5. Simple–Complex	2.7 (1.4)	3.2 (1.4)	3.4 (1.5)	3 (1.4)	4 (1.4)	3.4 (1.5)	3.6 (1.4)	3.4 (1.5)	3.2 (1.3)	2.9 (1.3)
6. Dull–Shiny	3.2 (1.2)	3.6 (1.2)	3.7 (1.2)	3.1 (1.3)	4.2 (1.2)	3.7 (1.4)	4.7 (1.2)	4.5 (1.4)	4.9 (1.4)	4.5 (1.5)
7. Small–Large	4.3 (1.1)	4.9 (1.3)	3 (1.2)	3.7 (1.4)	4.0 (1.3)	3.5 (1.1)	5.7 (0.7)	5 (1.4)	4.4 (1.3)	3.9 (1.3)
8. Unpleasant–Pleasant	3.7 (1.6)	4.2 (1.5)	4.8 (1.4)	4 (1.5)	5.3 (1.3)	4.8 (1.6)	4.6 (1.4)	4 (1.6)	3.9 (1.6)	3.4 (1.4)
9. Cool–Warm	3.6 (1.2)	3.8 (1.4)	4.7 (1.3)	4.4 (1.6)	5.1 (1.2)	4.6 (1.4)	3 (1.3)	2.7 (1.2)	2.9 (1.4)	2.8 (1.1)
10. Somber–Cheerful	3.4 (1.3)	4.2 (1.5)	3 (1.1)	2.8 (1.4)	4.2 (1.3)	3.9 (1.3)	4.4 (1.2)	3.9 (1.5)	4 (1.4)	3.5 (1.3)

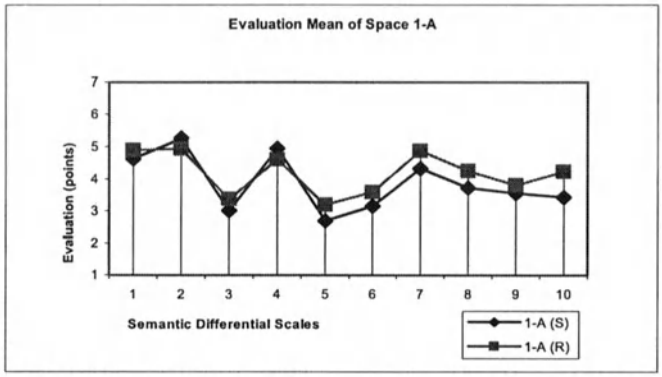


Figure 8. Evaluation means for scene 1-A
(S: actual scene, R: rendering, 1…10: scales according to table 3)

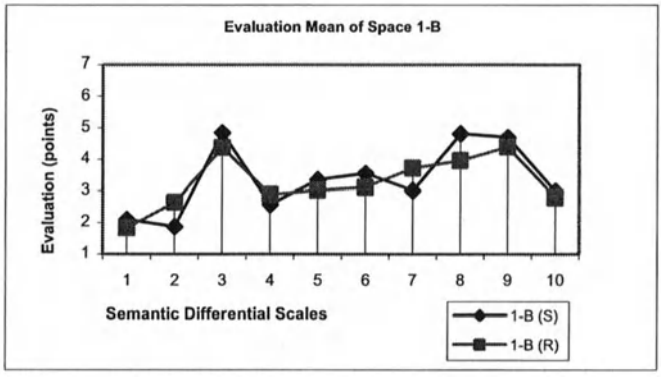


Figure 9. Evaluation means for scene 1-B
(S: actual scene, R: rendering, 1…10: scales according to table 3)

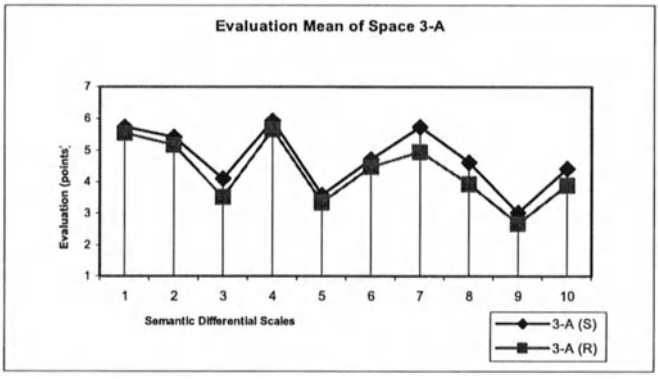


Figure 10. Evaluation means for scene 3-A
(S: actual scene, R: rendering, 1…10: scales according to table 3)

To establish an overall understanding of the degree of agreement between the results of the two tests, a regression analysis was performed (cp. figure 11) (McClave et al. 1997). The correlation (Multiple r) is found to be 0.91; the corresponding r2 of 0.83 indicates the variance accounted for by regression. Regression parameters are a slope of 0.81 (P-value 0.00) and intercept of 0.55 (P-value 0.02). These results imply a high level of congruence between impressions of the lighting gained from rendered images as compared to impressions gained from actual spaces, but with a relative reduction in the range of responses and constant offset of .55 for the rendered images.

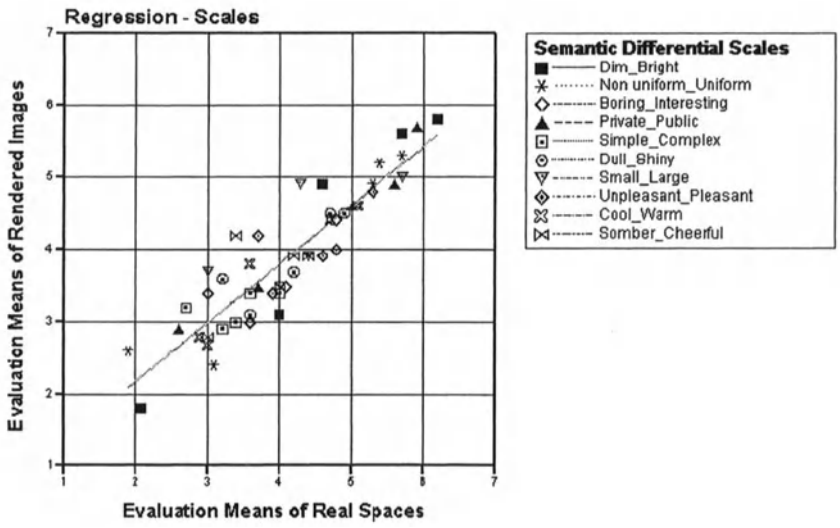


Figure 11. Regression analysis across all spaces and scales

Regression analysis was not only performed for the entire data, but also for individual scales. While the regression analysis of the entire data shows a high overall correlation, there are differences among individual scales in terms of their representational reliability (correlation) as well as in terms of their sensitivity to the stimuli variance (some scales display a wider range of numeric variability across different stimuli). The present study suggests that, while using rendered images to predict lighting quality features of real spaces, the dim/bright, non-uniform/uniform, boring/interesting, private/public, dull/shiny, and cool/warm scales are more reliable. Among this subset of scales, the dim/bright, non-uniform/uniform, and private/public display also a high sensitivity toward variance in stimuli.

In addition to regression analysis a non-parametric test (Kolmogorov-Smirnov) was used to compare the results of both evaluations. This test compares the evaluation medians for both data sets while considering each sample's distribution (Siegel and Castellan, 1988). The test was conducted for all 50 cases (10 scales and 5 lighting scenes). The resulting P-values of this test are summarized in table 4. Assuming a confidence level of 90% and a two-tailed test, P-values less than 0.05 would imply that there are significant differences between the evaluations of renderings versus actual spaces. As it can be seen from table 4, in 82% of the cases (41 out of 50), there is no significant difference between the subjective evaluations of actual scenes versus their rendered images.

Table 4. P-values of Kolmogorov-Smirnov test for all 50 cases, P-values less than 0.05 (in parenthesis) show significance at confidence level of 90% and two-tailed test.

Scene:	1-A	1-B	2	3-A	3-B
1. Dim–Bright	0.86	0.96	(0.00)	0.86	0.27
2. Non uniform–Uniform	0.39	(0.04)	0.11	1.00	0.54
3. Boring–Interesting	0.96	0.27	(0.04)	(0.04)	(0.01)
4. Private–Public	0.71	0.54	0.11	0.86	(0.02)
5. Simple–Complex	0.27	0.39	0.18	0.96	0.71
6. Dull–Shiny	0.27	0.54	0.54	0.71	0.71
7. Small–Large	(0.04)	0.07	0.39	0.07	0.18
8. Unpleasant–Pleasant	0.39	(0.01)	0.39	0.54	(0.04)
9. Cool–Warm	0.71	0.27	0.71	0.71	1.00
10. Somber–Cheerful	0.18	0.39	0.54	0.71	0.39

It is noteworthy that both the regression analysis and the Kolmogorov-Smirnov provide statistical evidence for the validity of the initial hypothesis.

4. CONCLUSION

The objective of the study described in the present paper was to empirically establish if and to what extent subjective lighting evaluation of lit architectural spaces can be reproduced using computationally rendered images of such spaces. For the sample of participants and lighting scenes tested in the study, the results suggest that such images can reliably represent certain aspects of the lighting conditions in real spaces. The overall correlation between the image-based and space-based evaluations was found to be significant. However, within the semantic differential metric, not all employed scales display both high representational consistency (from spaces

to their rendered images) and high sensitivity (to variances in the stimuli, i.e. differences in lighting-relevant attributes of various spaces).

Future research should further investigate potential effects of *a)* alternative semantic differentials, *b)* multiple space use types, *c)* different samples of test participants (in terms of age, profession, and cultural background), and *d)* dynamic experience of real spaces as opposed to evaluations based on a single vantage-point.

Moreover, it became clear in the course of the present study that multiple conditions must be met, if rendered images are to be used as reliable surrogates of real spaces for lighting design evaluation. Such conditions include, for example, *a)* the application of proper scales for the lighting design task at hand, *b)* proper construction of the computational model (geometry, material properties, photometric data, etc.), *c)* application of a scientifically sound and empirically validated visualization tool, *d)* proper calibration of hardware (i.e. particularly computer display unit) using one or more real spaces as reference.

5. REFERENCES

Flynn, J. E., T. J. Spencer, O. Martyniuk, and C. Hendrick, 1973, "Interim study of procedure for investigating the effect of light on impression and behavior", *Journal of Illuminating Engineering Society,* 3, p. 87-94.

Flynn, J. E., 1988, "Lighting-design decisions as interventions in human visual space". In: J. L. Nasar (Ed.) *Environment Aesthetics: Theory, research, & applications,* Cambridge University Press, New York, p. 156-169

Flynn, J. E., J. A. Kremers, A. W. Segil, and G. R Steffy, 1992, *Architectural interior systems: Lighting, acoustics, air conditioning (3rd ed.),* Van Nostrand Reinhold, New York.

Hendrick, C., O. Martyniuk, T. J. Spencer, and J. E. Flynn, 1977, "Procedures for investigating the effect of light on impression: Simulation of a real space by slides." *Environment and Behavior,* 9 (4), p. 491-510.

Lightscape, 1999, Autodesk release 3.2

McClave, J. T., F. H. Dietrich II and T. Sincich, 1997. *Statistics (7th ed.),* Prentice Hall, New Jersey.

Siegel, S., and N. J. Castellan, 1988, *Nonparametric statistics for the behavioral sciences (2nd ed.),* McGraw-Hill, New York.

Veitch, J. A. and G. R. Newsham, 1996, "Experts' quantitative and qualitative assessments of lighting quality", *The 1996 Annual Conference of the Illuminating Engineering Society of North America, Cleveland, OH, USA.*

Ensuring Usability of CAAD Systems
A Hybrid Approach

Sheng-Fen Chien
National Taiwan University of Science and Technology

Key words: Usability evaluation, GOMS analysis, Usability engineering, Object-oriented software engineering

Abstract: Many CAAD software prototypes today are developed with the aim to bring research results closer to practice. This paper describes a hybrid approach that integrates an Object-Oriented Software Engineering (OOSE) methodology with a usability analysis methodology—GOMS. This approach is examined through two case studies and has shown promising results. It enables CAAD system developers to be aware of usability issues and conduct usability evaluation as early as the analysis phase of the software development process. Consequently, this may improve the quality of CAAD software systems as well as ensure the usability of the systems.

1. INTRODUCTION

The advances of CAAD research have discovered many varieties of functionality a computer can provide to assist designers. To demonstrate the functionality and its usefulness, researchers develop working software prototypes and subject them to user evaluations. According to Nielsen (1993), "usefulness" deals with the question of whether a system can be used to accomplish desired goals, and it involves two issues—utility and usability: utility focuses on the functionality of a system, and usability examines how well users can use that functionality. For a simple prototype, its user interface may be adjusted quickly and as many times as necessary to achieve its intended functionality. For a prototype with complex functionality, however, its user interface design has to focus on ensuring usability so that its functionality can be evaluated systematically.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 361–374.
© 2001 *Kluwer Academic Publishers.*

In this paper, I present a software and usability engineering process that takes a hybrid approach to integrate the Object-Oriented Software Engineering (OOSE) methodology with the usability analysis methodology—GOMS. The objective of this hybrid approach is to enhance the evaluation phase of the software engineering process so that it covers the full range of the software lifecycle and ensures the usability of the CAAD software systems.

2. USABILITY AND SOFTWARE ENGINEERING

2.1 Usability Engineering: GOMS Analysis

Methods to evaluate usability can be categorized into two types—*usability analysis* and *usability testing*. Usability analysis evaluates a system based on knowledge derived from physical, psychological, or sociological theories, theories of design, or usability heuristics. Usability testing refers to the use of empirical investigation conducted with users of the system. Among usability analysis methods, the GOMS family of usability analysis techniques (John and Kieras, 1994) are based on the conceptual framework of human information processing (Newell and Simon, 1972) and aim at predicting the usability of a system before it has been tested. Moreover, GOMS analysis can be used early in the CAAD system development process to evaluate different ideas before the prototypes are implemented.

The general GOMS concept is to analyze knowledge of how to do a task in terms of the components of *goals*, *operators*, *methods*, and *selection rules*. Goals are what a user has to accomplish; they are often decomposed into subgoals, and all of the subgoals must be accomplished in order to achieve the overall goal. Operators are actions performed by the user in service of a goal; they can be perceptual, cognitive, or motor acts, or a composite of these. Methods describe procedures (sequences of operators and subgoals invocations) for accomplishing a goal. Selection rules determine which method to use when there is more than one method available to the user to accomplish a goal.

Figure 1 illustrates a GOMS model with annotations indicating how statements in the model relate to goals, operators, methods and selection rules. This sample model encodes the task of locating an "active node" in a "design space view". Additionally, the GOMS model shows that there are two alternative methods to view the design space: through a "2D tree view" or a "multi-layer view". Lastly, the model shows keystroke-level operations to accomplish a goal. If a designer should choose to view the design space

using the multi-layout view, the predicted task performance time could be estimated by adding the times needed to execute all related operators.

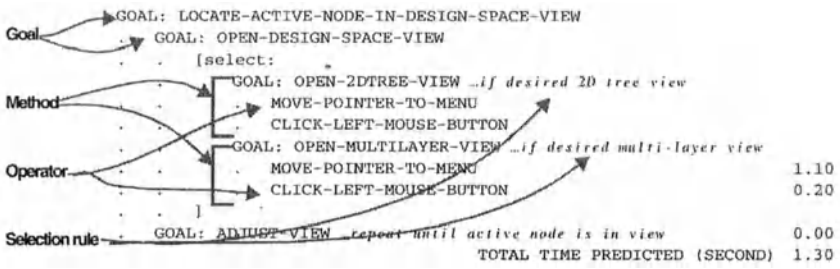


Figure 1. A sample GOMS model with time prediction

Constructing a GOMS model is usually time-consuming, especially when the task analysis process is performed as an extra effort aside from the software engineering process. This is where the hybrid approach comes in. In the following sections, I will illustrate how to take advantage of a OOSE use case description and expand it into a GOMS model.

2.2 Object-Oriented Software Engineering: A Use Case Driven Method

Developing a CAAD system is a complex task. OOSE methods attempt to handle the complexity in an organized way by working with models, each of which focuses on a certain aspect of the system. Coyne and his colleagues (1993) demonstrate the use of a particular OOSE method in the development of a design support system called SEED. This practice has enabled SEED researchers to assure the system's functionality through all development phases (for their continuing progress, see SEED, 2000). This particular method—a use case driven OOSE—emphasizes system development based on user requirements (Jacobson, Christerson, et al., 1992).

The core of this use case driven OOSE is a *use case model*. The model specifies a system's functionality from a user's perspective. It uses *actors* and *use cases* to define what exists outside the software system (i.e. actors) and what should be performed by the system (i.e. use cases). The actors represent everything that needs to exchange information with the system. They can be human users or other software systems. However, there is a clear distinction between actors and users—an actor represents only a certain role that a user can play. A use case is a specific way of using the system by an actor through a sequence of interactions to achieve a desired outcome.

For example, in the document of *SEED-Layout Requirement Analysis* (SEED, 1998), each use case is formulated in three parts: a *brief description* that summarizes the use case in a short sentence; a *flow of events* that describes the course of events during the performance of the use case; and a *comments section* for special considerations in the software development process regarding this use case. Using this format, a use case *Show Location of Active Node* can be documented as follows:

Show Location of Active Node

Brief Description:

The system displays a design space view and highlights the active node.

Flow of Events:

1. The designer requests display of a design space view.
2. The system displays a design space view with the active node highlighted.

Comments:

A node represents a *problem* or a *solution*. There may be more than one active node in a design space, i.e. an active problem and its associated active solution. A design space view does not necessarily display a complete view of all nodes populating the design space; it may be a partial view, e.g. a view of the problem decomposition hierarchy only. The highlighting mechanism may employ visual cues in addition to the color change method (e.g. reverse video display) commonly used.

This use case documentation is a base document throughout the software engineering process of a CAAD system. The functional requirement, the user interface design, and even GOMS models for usability evaluation will all be formulated based on this documentation.

2.3 Use Case Driven OOSE Meets GOMS Analysis

The GOMS analysis and the use case driven OOSE share a common point, i.e. both methods operate on a model of user tasks. The GOMS analysis relies on this model (GOMS model) to perform usability evaluations; the OOSE describes system behavior in the use case model, which defines how tasks are performed. Moreover, the GOMS model can be derived from the use case model if a goal-oriented approach (Cockburn, 1997) is taken to develop and structure use cases. My hybrid approach builds on this common base between the GOMS analysis and the use case driven OOSE, and introduces a process that refines the use case model into the GOMS model.

Let us review the use case *Show Location of Active Node* illustrated in Section 2.2. The complete course of events to perform this use case is described in its flow of events section as follows:

Show Location of Active Node

Flow of Events:

1. The designer requests display of a design space view.
2. The system displays a design space view with the active node highlighted.

Considering these two events in terms of user goals during the task, we can define the designer's top level goal:

GOAL: LOCATE-ACTIVE-NODE-IN-DESIGN-SPACE-VIEW

Naturally, if a design space view is available on the screen, the designer can go to that view directly and proceed to look for the active node. Conversely, the designer has to bring up the design space view first if it is not available on the screen. In either cases, the designer may have to adjust the display of the view in order to see the active node. Therefore, we can consider these two events as two subgoals to be achieved in order to accomplish the top level goal. Using GOMS notation, we write:

GOAL: LOCATE-ACTIVE-NODE-IN-DESIGN-SPACE-VIEW
. **GOAL:** OPEN-DESIGN-SPACE-VIEW *...if not available on screen*
. **GOAL:** ADJUST-VIEW *...repeat if active node not in view*

The indentation above indicates, for instance, that **GOAL:** ADJUST-VIEW is a subgoal of **GOAL:** LOCATE-ACTIVE-NODE-IN-DESIGN-SPACE-VIEW; and the text in italics says that this subgoal is to be invoked repeatedly until the active node appears in view. For comparison, Table 1 illustrates the mapping between the use case event flow and the GOMS model goal hierarchy discussed above.

Table 1. Mapping between use case flow of events and GOMS model goal statements

Show Location of Active Node	
Flow of Events	GOMS Model
1. The designer requests display of a design space view.	GOAL: LOCATE-ACTIVE-NODE-IN-DESIGN-SPACE-VIEW . GOAL: OPEN-DESIGN-SPACE-VIEW
2. The system displays a design space view with the active node highlighted.	. GOAL: ADJUST-VIEW ... *repeat if active node not in view*

The completion of GOMS models, however, requires detailed designs of the user interface in order to encode operators, methods, and selection rules. Consequently, the user interface can be evaluated by examining the appropriateness (from the cognitive aspect) of GOMS models and by

analyzing data collected from the user testing against GOMS models. For example, as the system development process continues, if a user interface design decision is made to offer the designer two alternative ways to view the design space (e.g. a 2D tree view and a multi-layer view), the subgoal can be expanded to model the choice between two methods:

GOAL: OPEN-DESIGN-SPACE-VIEW
. [select: **GOAL:** OPEN-2DTREE-VIEW ...*if desired 2D tree view*
. **GOAL:** OPEN-MULTILAYER-VIEW ...*if desired multi-layer view*]

The GOMS model can be continuously refined along the software engineering process with little or no extra effort. As more and more user interface design decisions are made (with or without a prototype or actual implementation), the GOMS model will have enough detail to predict the time needed to perform the particular task (e.g., see Figure 1 in Section 2.1).

3. CASE STUDIES

The hybrid approach has been experimented on two projects with promising results. The first project is SEED-Layout (Flemming and Chien, 1995). This approach is used to predict the performance of a newly added functionality. The predictions are compared with the empirical data collected from usability testing. In general, the empirical data agree with the predictions. The second project is a web environment that provides Feng-Shui recommendations for selecting and rearranging (the interior of) an apartment unit (Chien and Shih, 2000). The hybrid approach is used to evaluate user interface designs before the implementation.

3.1 SEED-Layout

SEED-Layout is a generative design system that supports schematic layout planning including the rapid generation of layout alternatives based on explicitly stated requirements (Flemming and Chien, 1995). In a SEED-Layout design session, a designer can explore alternative layout solutions, modify and refine problem specifications, *revisit* promising solutions, and perform many other operations in pursuit of a final layout solution for a design problem. A marker utility is planned and prototyped to provide assistance when a piece of information, such as a promising layout solution, is to be revisited for reasons such as reevaluation or comparison to other solutions. Two methods—GOMS analysis and user testing—are used to assess the marker utility. The two assessment methods are based on the scenario described above: they compare the times a designer may take to visit (first visit) and revisit (second visit) a layout solution in different

treatments—without the marker utility or with variations of the utility—of the SEED-Layout environment.

The relevant GOMS models are obtained as a result of the hybrid approach used to develop the marker utility in SEED-Layout (see Appendix A for a sample documentation). This analysis examines the user performance in two tasks:

1. to locate target layouts, and
2. to revisit the target layouts.

The detail GOMS analysis examines user performance in these two tasks in three different SEED-Layout user interface treatments: Control Treatment (without marker utility), Map-and-Notepad Treatment (with markers and marker list), and Map Treatment (with markers only).

The GOMS analysis for the particular task of visiting a specific target layout predicts that having marker support reduces the time a designer takes to revisit that layout. The task performance time could be further reduced if the marker support provides, in addition to visual cues on the display, a marker management utility such as the marker list in one the SEED-Layout user interface treatment.

The three user interface treatments are implemented (see Appendix B for snapshots) and evaluated through empirical studies with two designers to perform the same tasks describe above. A quantitative data analysis based on the keystroke events recorded during each task was performed. The result supports the prediction from the GOMS analysis. (For details of this case study, please refer to Chien, 1998.)

In this case study, the hybrid approach enables SEED-Layout developers to examine the user interface design of a newly planned functionality and predict the performance.

3.2 WIDE-Kindom

Most apartment units in Taiwan are sold before they are ever been built. Therefore, apartment buyers are able to customize the interior of their units (e.g., by changing finishes, fixtures, or layout arrangements) before the building construction begins. WIDE-Kindom is a web environment geared to support this customization process. Among its services, WIDE-Kindom provides "Feng-Shui recommendations" as an alternative to help buyers identify suitable apartment units, as well as fine-tuning apartment layout to achieve a harmonious living environment.

Feng-Shui is a set of rules developed by ancient Chinese to relate people and the man-made environment to the natural environment (for further information regarding Feng-Shui, see Walters, 1991). In Taiwan and Hong Kong, many people arrange their living and working environment according

to Feng-Shui. Main entrance, stove, toilet, and bed are four key elements in Feng-Shui. The "location" and "orientation" of each element in a house asserts energy that contributes to either the harmony or the conflict between a resident and the house. On the other hand, people are not all the same; they belong to one of eight different types according to their birthday and time.

WIDE-Kindom developers formulate the use case for providing Feng-Shui recommendations as follows (note that the use case is written from an actor's perspective):

Obtain Feng-Shui Recommendations

Brief Description:

The system provides Feng-Shui recommendations according to user's birth information and apartment layout.

Flow of Events:

1. The user requests Feng-Shui recommendations.
2. The system requests input for birth information.
3. The system displays apartment layout with recommendations.

Comments:

This use case should be further elaborated into three supporting use cases: *Input Birth Information*, *Show Available Apartments*, and *Show Feng-Shui Recommendations*.

Throughout the software engineering process, use cases are refined and elaborated. Given the complexity of this task, the developers have noted in the comments section that three supporting use cases should be formulated to accomplish the task. As a result, this forms a hierarchical structure of use cases, which is comparable to the goal hierarchy in a GOMS model. In other words, a GOMS model can be easily formulated as follows:

GOAL: OBTAIN-FENG-SHUI-RECOMMENDATIONS
. **GOAL:** INPUT-BIRTH-INFO
. **GOAL:** SELECT-APARTMENT
. . **GOAL:** VIEW-AVAILABLE-APARTMENTS *...repeat if desired one not in view*
. . **GOAL:** IDENTIFY-APARTMENT
. **GOAL:** VIEW-RECOMMENDATIONS
. . **GOAL:** ADJUST-APARTMENT-LAYOUT *...repeat until satisfied*
. . VERIFY-RECOMMENDATIONS

As the software engineering process continues, one of the supporting use case *Show Feng-Shui Recommendations* are described as follows:

Show Feng-Shui Recommendations

Brief Description:

The system displays Feng-Shui recommendations.

Flow of Events:

1. The user identifies an apartment layout.
2. The system displays the apartment layout with recommendations.

Comments:

Note that the "recommendations" may be provided in various forms, for example flashing icons to show conflicts or scores for overall layouts.

From here on, WIDE-Kindom developers put their focuses on the user interactions and begin to sketch out possible interface designs. Since the Feng-Shui recommendations service hopes to assist users in customizing the interior of an apartment unit, a decision is make that this service should be interactive and provide recommendations upon each user adjustment. Furthermore, a user adjustment can be made through the four Feng-Shui elements: main entrance, stove, toilet, and bed. Two alternative interaction schemes are discussed: upon the user selection of apartment layout, the system displays an initial recommendation based on the user's birth information; or the user starts with no initial recommendation, and goes on to get the recommendation by adding/adjusting elements on the layout. For comparison, these considerations are encoded in GOMS models shown on Table 2.

Table 2. Two GOMS models for the "View Recommendations" task

Interaction A: no initial recommendation	Interaction B: with initial recommendation
GOAL: VIEW-RECOMMENDATIONS	**GOAL:** VIEW-RECOMMENDATIONS
. **GOAL:** ADJUST-APARTMENT-LAYOUT	. VERIFY-RECOMMENDATIONS
...repeat until satisfied	. **GOAL:** ADJUST-APARTMENT-LAYOUT
. . RECALL-RECOMMENDATIONS	*...repeat if not satisfied*
. . [SELECT	. . RECALL-RECOMMENDATIONS
. . GOAL: ADJUST-MAIN-ENTRANCE	. . [SELECT
. . GOAL: ADJUST-BED	. . GOAL: ADJUST-MAIN-ENTRANCE
. . GOAL: ADJUST-STOVE	. . GOAL: ADJUST-BED
. . GOAL: ADJUST-TOILET]	. . GOAL: ADJUST-STOVE
. . COMPARE-RECOMMENDATIONS	. . GOAL: ADJUST-TOILET]
. VERIFY-RECOMMENDATIONS	. . COMPARE-RECOMMENDATIONS

Having done these analyses, WIDE-Kindom developers implement prototypes with different initial recommendation settings and with varying degrees of freedom for element adjustments (see Appendix C for sample prototypes). The resulting system is publicly accessible and most users appear to enjoy interacting with it.

4. CONCLUSION

The experience gained from two case studies indicates that the hybrid approach enables CAAD system developers to be aware of usability issues and conduct usability evaluation as early as the analysis phase of the

software development process. Consequently, this may improve the quality of CAAD software systems as well as ensure the usability of the systems.

For small proof-of-concept prototypes, this approach may not be suitable since conducting direct usability testing may be an easier and more effective method to ensure usability. However, many CAAD software prototypes today are developed with the aim to bring research results closer to practice. To achieve this objective, rigorous methods are needed. The hybrid approach provides just that.

A reviewer commented "[d]ifferent designers will use CAAD systems in different ways, each reflecting the nature of their own design practice." User modeling is difficult. However, through explicitly modeling the behavior of a CAAD system and its users, the CAAD researchers may obtain deeper understanding of the system, the users and interaction between these two (Coyne, Flemming, et al., 1993). The hybrid approach is developed with that in mind. What design drawings and models have done for architects is what the hybrid approach of usability and software engineering hope to do for CAAD system developers.

5. ACKNOWLEDGEMENTS

The SEED-Layout case study was conducted in Carnegie Mellon University with the assistance of the SEED research team, especially Professor Ulrich Flemming. The WIDE-Kindom case study is sponsored by Kindom Construction Co, Taiwan.

6. REFERENCES

Chien, S.-F., 1998, "Supporting Information Navigation in Generative Design Systems", PhD Dissertation, School of Architecture, Carnegie Mellon University, Pittsburgh, PA.

Chien, S.-F. and S.-G. Shih, 2000, "A Web Environment to Support User Participation in the Development of Apartment Buildings", in: *Special Focus Symposium on WWW as the Framework for Collaboration, InterSymp 2000*, July 31-August 5, Baden-Baden, Germany, p.225~231.

Cockburn, A., 1997, "Using Goal-Based Use Cases", *Journal of Object-Oriented Programming*, 10(7), p. 56-62.

Coyne, R.F., U. Flemming, P. Piela, and R. Woodbury, 1993, "Behavior Modeling in Design System Development", in: Flemming and Van Wyk (eds.), *CAAD Futures '93*. Elsevier Science Publishers, Amsterdam, p. 355-354.

Flemming, U. and S.-F. Chien, 1995, "Schematic Layout Design in SEED Environment", *Journal of Architectural Engineering*, 1(4), p. 162-169.

Jacobson, I., M. Christerson, P. Jonesson, and G. Overgaard, 1992, *Object-Oriented Software Engineering: A Use Case Driven Approach*, Addison-Wesley, Reading, MA.

John, B.E. and D.E. Kieras, 1994, "The GOMS Family of Analysis Techniques: Tools for Design and Evaluation", Technical Report CMU-CS-94-181/CMU-HCII-94-106, School of Computer Science, Carnegie Mellon University.
Newell, A. and H.A. Simon, 1972, *Human Problem Solving*, Prentice-Hall, Englewood Cliffs.
Nielsen, J., 1993, *Usability Engineering*, AP Professional, Cambridge, MA.
SEED, 1998, "SEED-Layout Requirement Analysis", URL: http://seed.edrc.cmu.edu/SL/SL-start.book.html.
SEED, 2000, "SEED: Project Web Site", URL: http://seed.edrc.cmu.edu.
Walters, D., 1991, *The Feng-Shui Handbook*, Hopper-Collins, London.

7. APPENDICES

Appendix A: A sample SEED-Layout Requirements Analysis document.

Go To Marked Node

Brief Description

The system activates and displays a node selected by the designer from among those that are marked.

Flow of Events

1. The designer selects a marker from a design space view and requests its activation.
2. The system activates the node associated with the selected marker.
3. If the node is a solution, the system activates and displays the solution and its associated problem. If the node is a problem, the system activates and displays the problem and its *last active* solution.

Interaction Diagram

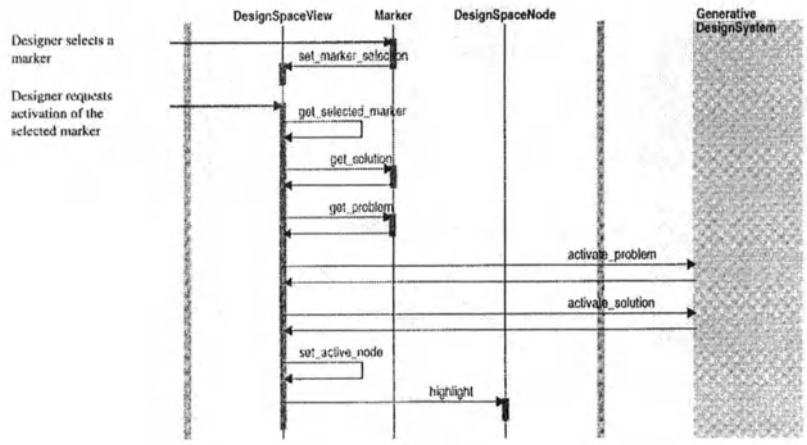


Figure A-1. Use case document: Go To Marked Node (1)

User Interface

1. The designer selects (i.e. clicks the **<Left-Mouse-Button>** on) a marker in a design space view; the system highlights the selected marker (for example, see Figure 6).
2. The designer presses down the **<Right-Mouse-Button>** on the selected marker.
3. The system brings up a pop-up operation menu.
4. The designer selects (i.e. releases the **<Right-Mouse-Button>** on) the **Go To** option.
5. The SEED-Layout system updates the DW with the proper layout; if necessary, it refreshes the PSW with the information of the associated problem specification, and properly refreshes the PHW and DSW to display active problem specification and layout.

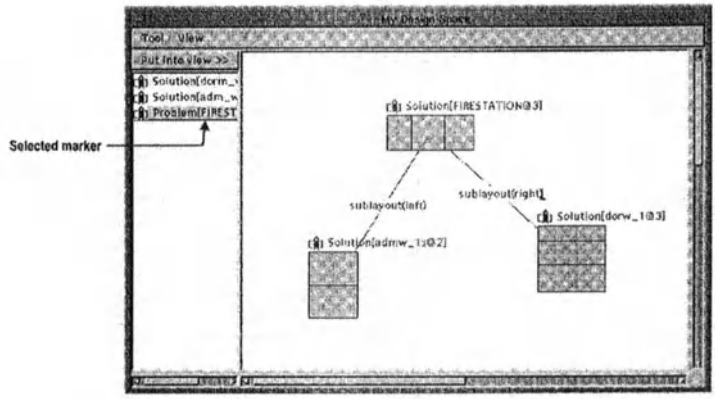


Figure 6. A design space view that shows marked nodes and markers

GOMS Model

```
GOAL: GO-TO-MARKED-NODE
.   GOAL: SELECT-MARKER
.   .   MOVE-POINTER-TO-MARKER
.   .   CLICK-LEFT-MOUSE-BUTTON
.   .   VERIFY-SELECTION
.   GOAL: ISSUE-GO-TO-COMMAND
.   .   MOVE-POINTER-TO-SELECTED-MARKER
.   .   PRESS-RIGHT-MOUSE-BUTTON
.   .   MOVE-MOUSE-TO-GO-TO
.   .   VERIFY-HIGHLIGHT
.   .   RELEASE-MOUSE-BUTTON
.   VERIFY-DISPLAYS
```

Figure A-2. Use case document: Go To Marked Node (2)

Appendix B: Sample screen snapshots for various SEED-Layout user interface treatments.

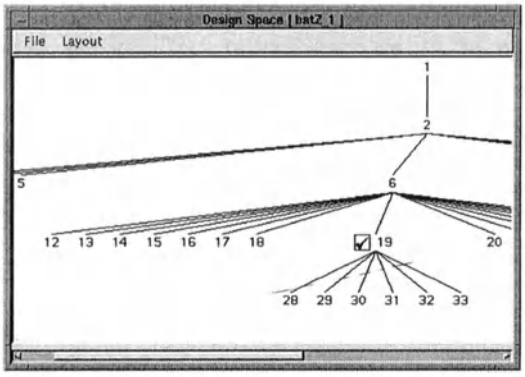


Figure B-1. Control Treatment

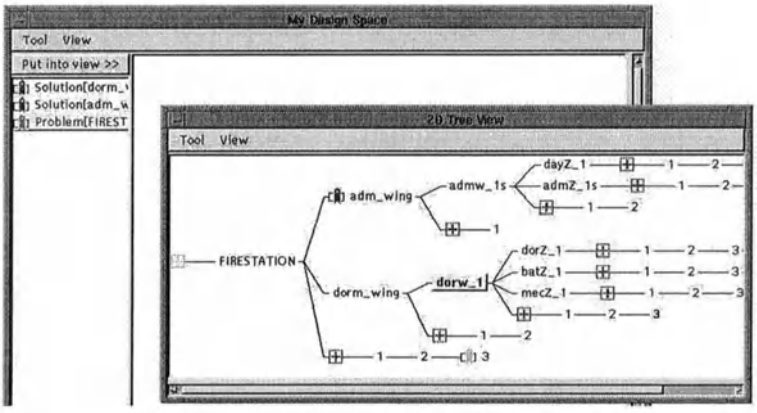


Figure B-2. Map-and-Notepad Treatment

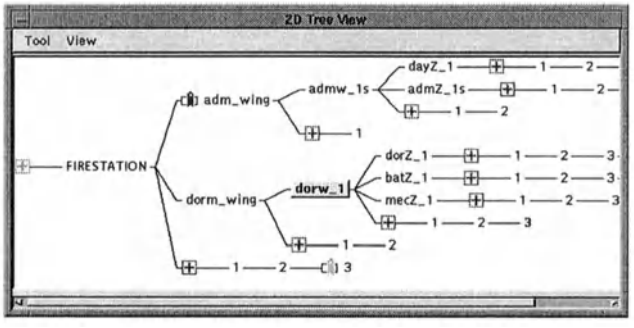


Figure B-3. Map Treatment

Appendix C: Prototypes of WIDE-Kindom Feng-Shui module.

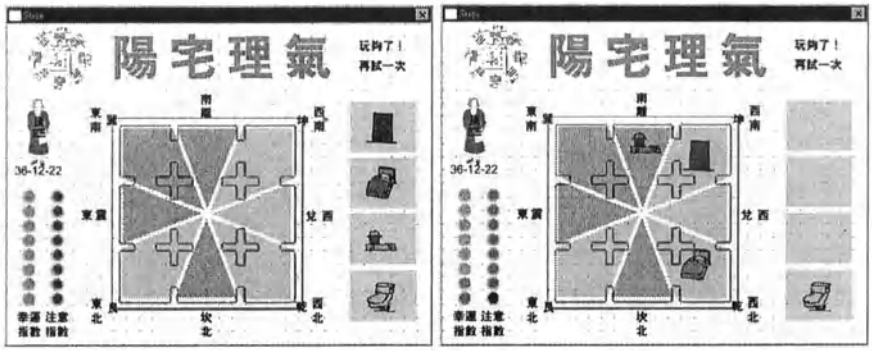


Figure C-1. Version A: no initial recommendations/limited element adjustment

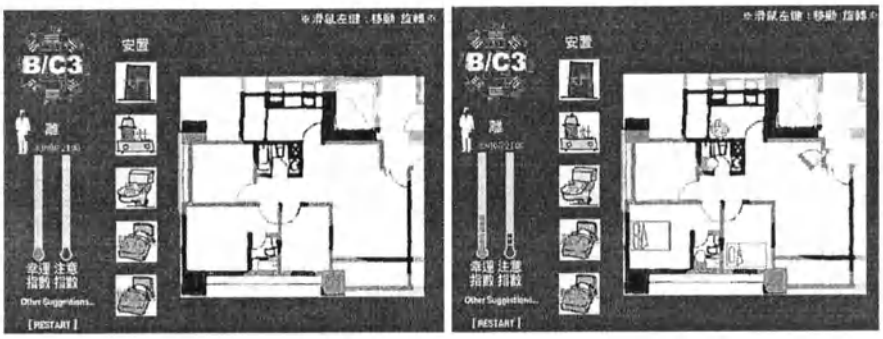


Figure C-2. Version B: no initial recommendations/unlimited element adjustment

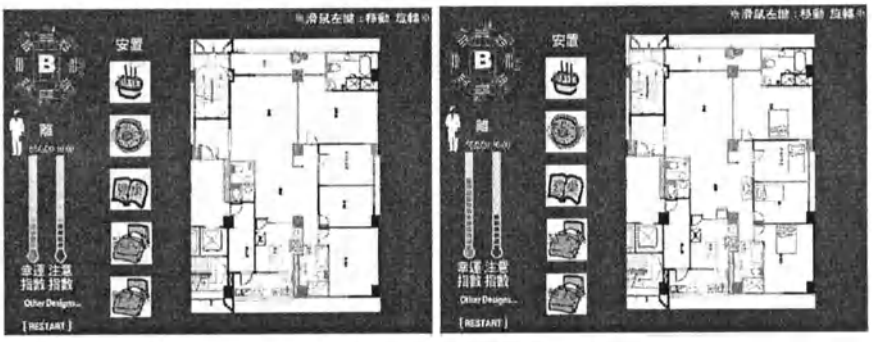


Figure C-3. Version C: initial recommendations/limited element adjustment

Tolerating Inconsistencies

The Distributed Perspectives Model

Oliver Hoffmann, Markus Stumptner and Talik Chalabi
Technical University of Vienna

Key words: CAD, Microstation, artificial intelligence, creativity, urban design, typology, Java, JATLite, JATLiteBean, agent, JESS

Abstract: A new design model is presented. Information on the design is distributed over multiple self-contained design perspectives and translation functions between design perspectives. Inconsistencies between specifications in different design perspectives introduced by human designers are temporarily tolerated in order to support creative design processes. The implementation of a design support system currently under evaluation is outlined.

1. MOTIVATION

Tolerating inconsistencies during the design process can be seen as a necessary prerequisite for creativity (Hoffmann, Stumptner, et al, 2000). Architectural design in particular deals with both "measurable" criteria like economy, building codes, environmental issues, etc. and "unmeasurable" factors like cultural values, perception, psychology, *Zeitgeist*, esthetic canons, etc. Therefore, computer support for design should provide for heterogeneous representation of design problems and admit modifications of any assumptions at any stage of the design process. Instead of adopting a view of the design process as a linear sequence of design steps with limited revisions of the design goals, we advocate a view characterized by intertwining of design specifications and design requirement changes, which are possibly executed by different individuals.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 375–386.
© 2001 *Kluwer Academic Publishers.*

2. AN URBAN DESIGN EXAMPLE

Urban design is a creative task that demands the juggling of issues from multiple domains such as architecture, zoning laws and traffic planning. The urban designer constantly alters his point of view and frequently re-evaluates his precepts to arrive at better or innovative solutions. As an example, the task to create an urban design for a rectangular residential area with a certain required built up density is investigated (density is defined as the ratio of built up space to plot area). The designer first chooses an existing building type to work with.

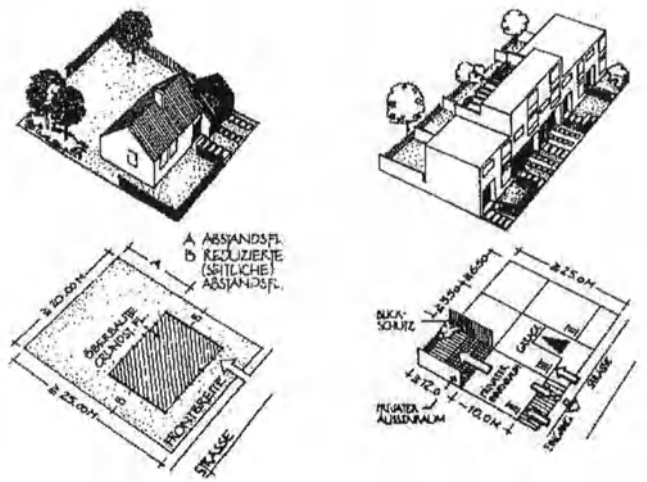


Figure 1. Residential building typology

In our example the designer uses two standard residential building types: single family house and row house (fig. 1) (Prinz, 1995). Both types have requirements related to building size and setback from neighboring plots and some key data associated with them, such as built-up density. In this example, the designer initially chooses to work with single-family houses (left side of fig. 1). This type allows a relatively high degree of flexibility in the two-dimensional (2D) layout. However, when the minimum setback distances for single family houses is taken into account, the required built-up density cannot be achieved. In the next step, the designer changes the selected building type to row house (right side of fig. 1) which helps to fulfil the density requirements but does not allow enough flexibility in the overall design. Therefore, the designer chooses to ignore the limitations of existing typologies for the time being and shifts individual floors by 90-degree, which results in the creation of a new building type based on the underlying row house. Fig. 2 shows an overall view of the resulting design. Fig. 3 shows the detailed facade of the first "row" exhibiting the variety of solutions attained.

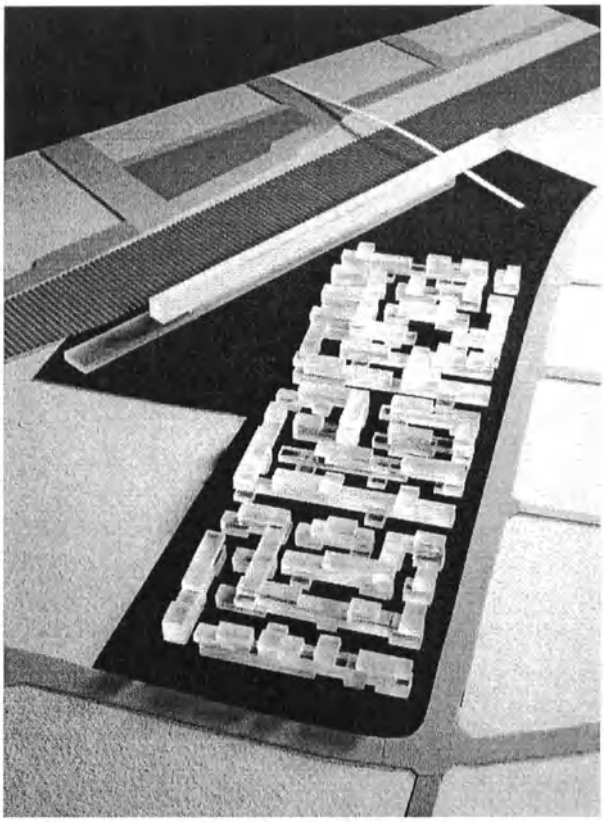


Figure 2. Result of creative transformation of row house typology

In effect, the designer has changed the rules for residential building types while using them in a specific design. Therefore, the designer has introduced a high degree of inconsistency between existing building types and the way the building types were used: From the computer support point of view, the 90° turn of 1st floors against ground floors is inconsistent with the row house type, since the row house is based on the linear addition of identical units. From the designer's point of view, the adaptation of existing typologies to site-specific design does not constitute inconsistency but an integral part of creative designing. The inconsistencies are resolved at the end of the design process, with the definition of a new building type. Computer support for this kind of creative design process has to tolerate inconsistencies at certain design stages.

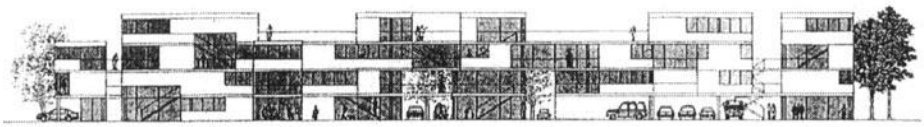


Figure 3. A new building type

3. DESIGN PERSPECTIVES

To allow designers to "change their perspective" on a given design, we introduce the concept of design perspectives. A design perspective is not a subset of a central data source, but a self-contained view of the design with a set of design requirements and specifications. The term "perspective" is used in the sense of "symbolic form" (Panofsky, 1924).

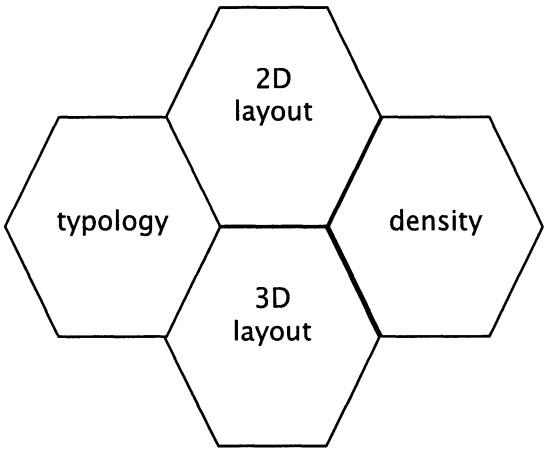


Figure 4. Design example with four perspectives

For the sake of simplicity, we restrict the design perspectives in our example to a small number of "measurables", all of them explicitly modeled in a design support system (fig. 4): typology (referring to the type of house and its relation to the ground it is built on), 2D layout (referring to the street grid), 3D layout (buildings as arranged on the site) and density (the urban density achieved). The term "layout" is used more in a computer science sense here, that is, some arrangement of shapes irrespective of their architectural interpretation.

At a given point in the design process, the current state of the design object, i.e., the specifications produced by the designer in a particular perspective have to be consistent with requirements in the same perspective. The current state in one perspective generally cannot be checked against requirements in other perspectives, since the symbols used in one

perspective might have a different interpretation in another perspective. For example, the symbol "building" could at a given point be interpreted as "2-dimensional closed multi-line shape" in the 2D layout perspective and as "set of solid objects in direct contact with label 'building'" in the 3D layout perspective.

4. TRANSLATIONS

Perspectives are connected using translation functions. A translation function A => B is a mapping from specifications in perspective A, i.e., the current state of the design object in that perspective as specified by the designer, to requirements in perspective B. The fact that specifications are translated into requirements rather than specifications is the primary distinguishing factor to approaches like the DESIRE system (Brazier, Langen, et al, 1994). The design is globally consistent at a given point in the design process if specifications in design perspectives are consistent with requirements from within the perspectives (internal consistency) and requirements resulting from translations (external consistency). A translation is similar to a filter as defined in filter mediated design (Haymaker, Ackermann, et al, 2000), in that a translation, like a filter, transforms information on the design from one representation to another. But in contrast to filter mediated design, there is no central representation in the distributed perspectives model. There can also be bidirectional translations between perspectives. Moreover, translations, as well as perspectives, can be added to and removed from the system at any point in the design process. In particular, different versions of a translation from one perspective to another might be available, and these different versions can be included in or excluded from the system at any point in the design process. Fig. 5 shows a possible set of translations between perspectives for the presented urban design example. For instance, a selection of a specific grid in the 2D layout perspective can be translated into requirements for the 3D layout: If the 3D layout is to be externally consistent with the 2D grid selected, buildings in the 3D perspective have to be arranged according to the 2D grid. Similarly, specifications from the density perspective can be translated into requirements for the 3D layout: If 3D layout is to be externally consistent with the density perspective, 3D layout has to achieve a certain density. In this example, there are 2 translations between the typology perspective and the 3D layout perspective. Therefore, it is possible both to translate specifications for building types into requirements for buildings in the 3D

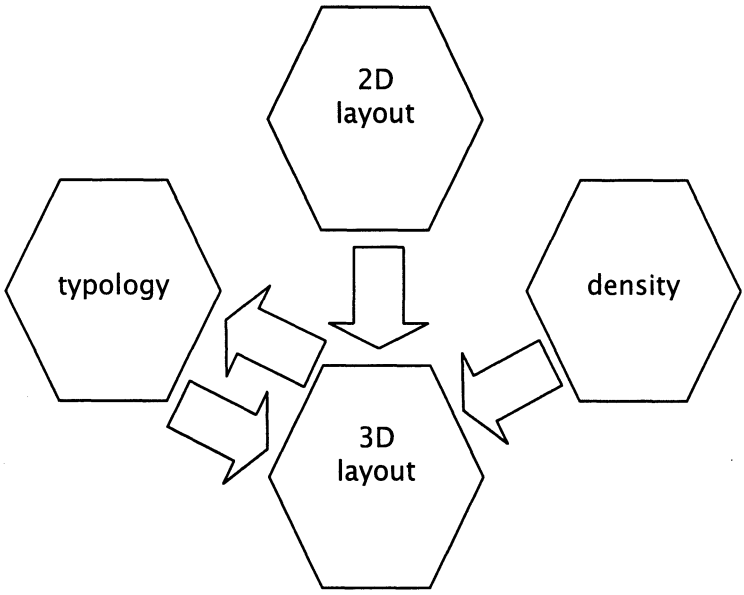


Figure 5. Translation functions between design perspectives

layout and to translate given buildings in the 3D layout into prototypes for new or updated building types.

5. IMPLEMENTATION

The distributed perspectives prototype has been implemented on the basis of a multi-agent system. Agent-based programming was applied more in the sense of software engineering (Petrie, 2000) than as an implementation of the belief-desire-intention theory (Rao and Georgeff, 1995). The human designer performs changes in some of the available perspectives and agents are used for translating between perspectives. Therefore, there is no control structure for designing (the human designer is free to choose among possible design steps).

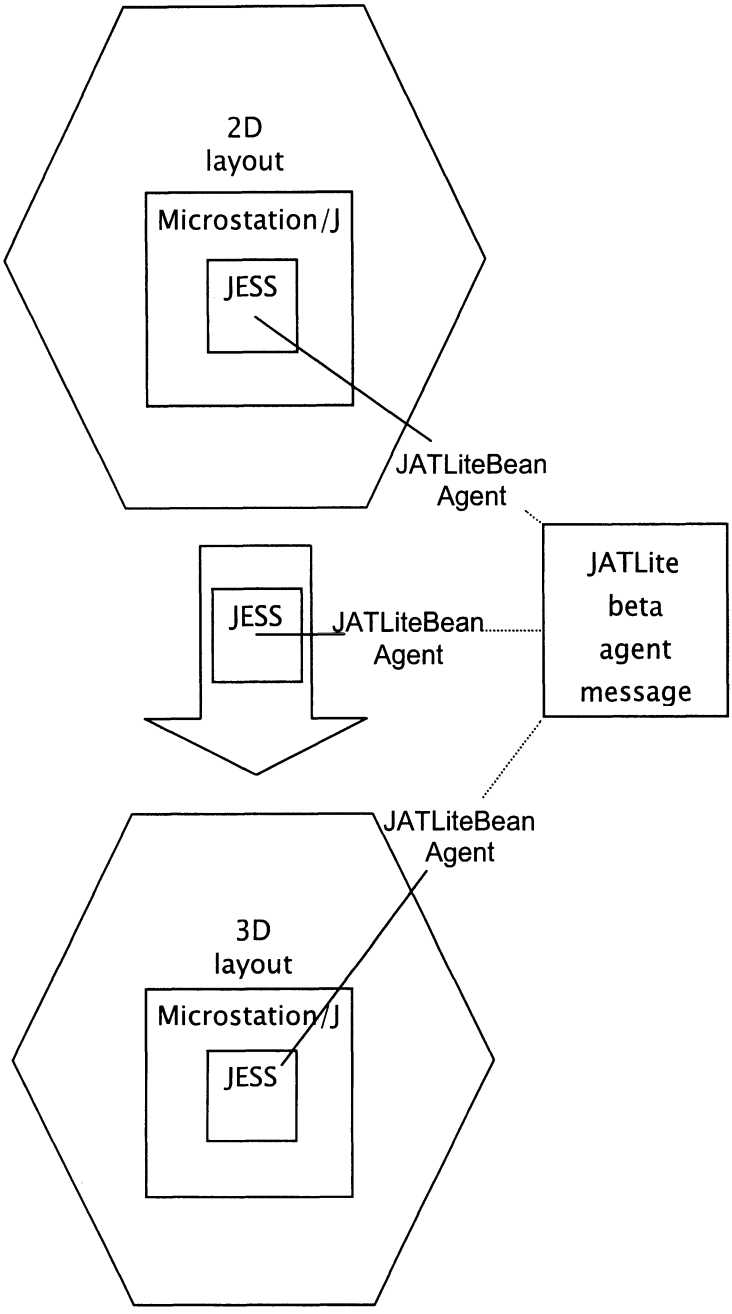


Figure 6. translation function implementation

Fig. 6 shows the software architecture chosen on the example of a translation function from the 2D layout perspective to the 3D layout perspective. The 2D layout perspective is implemented as an extension of

the commercial CAD software Microstation/J (Bentley, 2000). Design support is provided through one or many design agents implemented in the Java expert system shell or JESS (Sandia, 2000). JESS is an implementation of the Rete algorithm (Forgy, 1982) in Java. Communication between perspectives and translation functions is implemented through one or many JATLiteBean agents. JATLiteBean (Moreale and Kinlay, 2000) is an extension of the Java agent template JATLite (Jeon, Petrie, et al, 2000).

Translation of the 2D layout specifications into 3D layout requirements is provided in the following fashion: If a new or updated grid is selected in the 2D layout perspective, the JATLiteBean agent sends a message to all relevant translation functions containing the updated specifications. The JESS system implementing the translation function from the 2D layout to 3D layout receives this message, translates the 2D grid specifications to 3D layout requirements and adds rules for achieving these requirements. Then the translation function JESS system sends a message to any relevant perspectives containing new rules for the 3D layout through a JATLiteBean agent. The 3D layout perspective receives this message through a JATLiteBean agent and a design support agent incorporates the new rules into its rule base. Depending on the kind of translation performed, the new rules can be used for design automation or design critique.

6. DESIGN PROCESS

The design process can go through different stages with different characteristics. These stages do not occur in a pre-determined sequence, instead transition from one stage to another is introduced by human designers interacting with the design support system. Some examples of design stages are described below.

6.1 Problem-Solving Stage

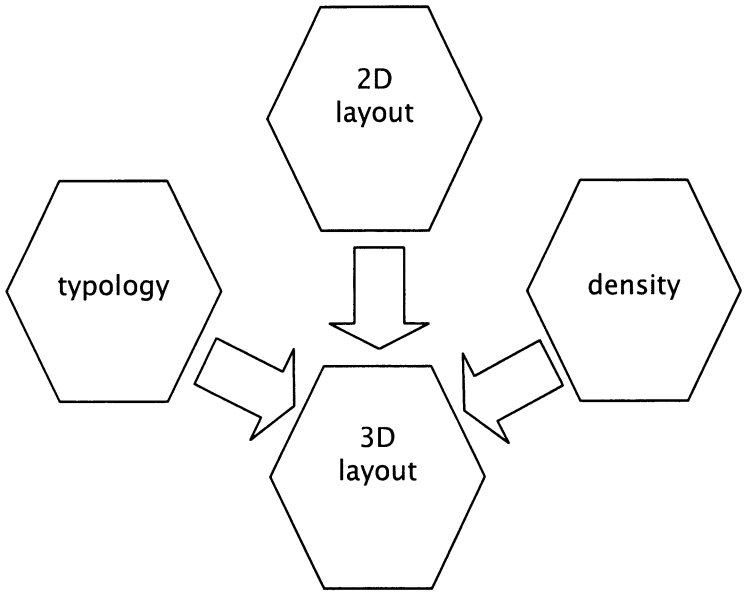


Figure 7. Problem-Solving Stage

Design as problem solving can be seen as the classical application of artificial intelligence in design (Brown and Chandrasekaran, 1989) In the distributed perspectives model, design as a problem-solving activity occurs under the following conditions:

- Exactly one design perspective is dynamic, that is, changes in design specifications are only occurring in this one design perspective.
- All specifications in other design perspectives have been successfully translated into requirements for the dynamic perspective.

The human designer and/or design support agents in the dynamic perspective try to find specifications that are consistent with the requirements. In the urban design example, the design problem is placing buildings in the 3D layout in such a way that density, 2D layout (grid) and building type requirements are met.

6.2 Creative Transformation

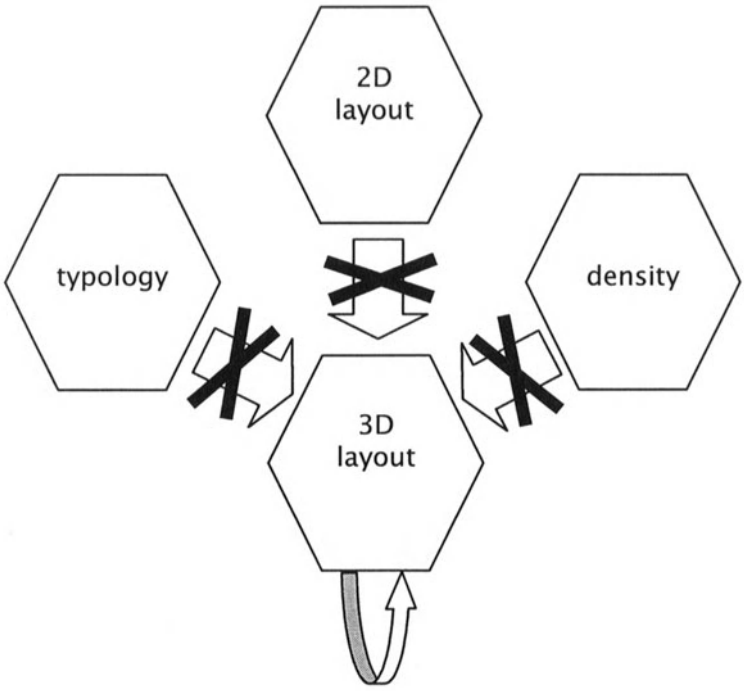


Figure 8. Creative Transformation

A creative design transformation can occur under the following conditions:

- All design perspectives contain design specifications.
- Translation functions are temporarily turned off.
- Specifications in one of the perspectives are modified in order to create an innovative design.

The creative transformation stage offers the opportunity to escape the "design as mapping" dilemma (Tomiyama, 1994). Fluctuating information, that is, temporary data that were produced on the basis of previous design knowledge (Oeser and Seitelberger, 1995); (Hoffmann and Kollingbaum, 1996), is modified to form the basis of a creative extension of the design problem space (McLaughlin, 1993). In the urban design example the row house building type is transformed *in situ* into a new building type.

6.3 Learning

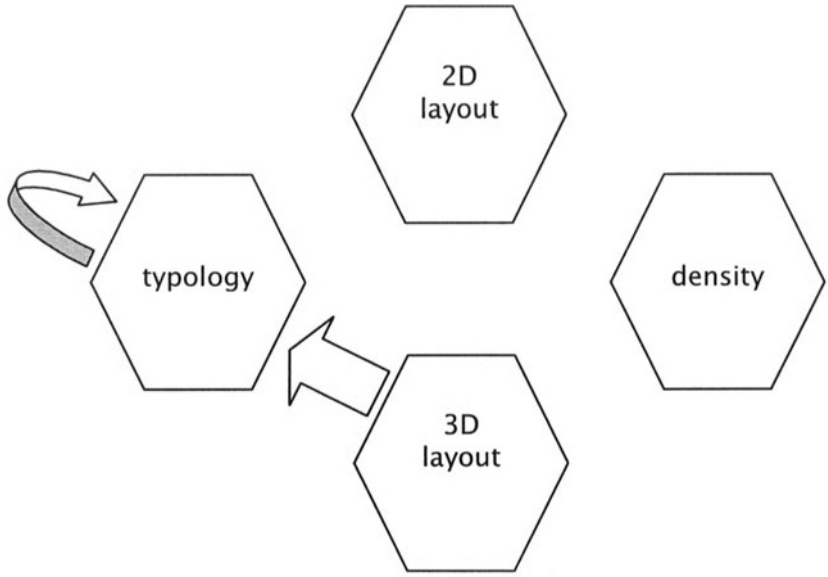


Figure 9. Learning

Learning as a result of a creative transformation occurs under the following conditions:

- A design perspective has received requirements resulting from specifications in a previous design stage.
- The flow of translation is reversed in order to adapt the set of possible specifications.

In the urban design example the new building type in the 3D perspective is translated into requirements for a new building type in the typology perspective. The human designer adapts the typology accordingly.

7. CONCLUSION AND OUTLOOK

In this paper, we have presented a new design model for creative design support, based on the notion of perspectives, different aspects of a design which provide separate and self-contained views of the design object. The interaction between perspectives happens in terms of transformations that map the current description in one perspective into requirements for the other perspectives. Contrary to classical views of design support, we show that tolerating inconsistencies between perspectives can in fact lead to better

support of creative design processes. We have described the applicability of the design model to different kinds of design support and have outlined a prototype implementation. The perspective-oriented problem representation is naturally suited to an agent-based implementation approach. Usability of the prototype will be tested by urban design experiments involving designers with different levels of expertise.

8. REFERENCES

Bentley, 2000, http://www.bentley.com/products/mstation/j/

Brazier, F. T. M., P. H. G. Langen, Zs. Ruttkay and J. Treur, 1994 "On Formal Specification of Design Tasks", J.Gero, F. Sudweeks (eds.), *Artificial Intelligence in Design '94,* Kluwer, Dortrecht, The Netherlands, p. 535-552.

Brown, D. C. and B. Chandrasekaran, 1989 *Design Problem Solving: Knowledge Structures and Control Strategies*, Morgan Kaufmann, San Mateo, CA, USA.

Forgy, C. L., 1982 "Rete: A Fast Algorithm for the Many Pattern/ Many Object Pattern Match Problem", *Artificial Intelligence* 19 (1982), pp. 17-37

Haymaker, J., E. Ackermann and M. Fischer, 2000 "Meaning Mediated Mechanism, Prototype for constructing and negotiating meaning in collaborative design", J. Gero (ed.), *Artificial Intelligence in Design '00*, , Kluwer, Dortrecht, The Netherlands, p. 691-715.

Hoffmann, O. and M. Kollingbaum, 1996 "Creativity as Transformation: Multi-Agent Systems and Human Cognition", *Creativity and Cognition 2*, Loughborough University, Loughborough, England, UK, p. 19-26.

Hoffmann, O., M. Stumptner and T. Chalabi, 2000 "Consistency, Creativity and Perspectives", *Artificial Intelligence in Design '00*, workshop notes on Developing Intelligent Support for Collaboration in Distributed Design, Worcester, MA, USA.

Jeon, H., Petrie C. and M. R. Cutkosky, 2000 "JAT*Lite*: A Java Agent Infrastructure with Message Routing", *IEEE Internet Computing*, 4(2), (March/April 2000), p.87-96.

McLaughlin S., 1993 "Emergent Value in Creative Products: Some Implications for Creative Processes". *Modeling Creativity and Knowledge-Based Creative Design*, Erlbaum Associates, Hillsdale, NJ, USA.

Moreale, E. and B. Kinlay, 2000, JATLiteBean, http://waitaki.otago.ac.nz/JATLiteBean/

Oeser E. and F. Seitelberger, 1995 "*Gehirn, Bewußtsein und Erkenntnis*", Wissenschaftliche Buchgesellschaft, Darmstadt, Deutschland.

Panofsky, E., 1924, "Die Perspektive als symbolische Form," *Vorträge der Bibliothek Warburg,* Leipzig, Deutsches Reich.

Petrie C., 2000, "Agent-Based Software Engineering", Invited talk, *Fifth International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, Manchester, England, U.K.

Prinz, D., 1995 *Städtebau* volume 1, Köhlhammer, Stuttgart, Deutschland.

Rao, A. S. and M. P. Georgeff, 1995 "BDI agents: From theory to practice," Technical Report 56, Australian Artificial Intelligence Institute, Melbourne, Australia.

Sandia, 2000, http://herzberg.ca.sandia.gov/jess

Tomiyama T., 1994 "From General Design Theory to Knowledge-intensive Engineering", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 8(4) (1994), pp.319-333.

Urban-CAD, A Design Application for Urbanism

D. Chitchian[1,2], E.G.M. Sauren[1], J. Heeling[1]
[1]*Delft University of Technology*
[2]*Iran University of Science and Technology*

Key words: Urban design process, Scaling, Abstracted and detailed levels, CAAD

Abstract: The existing CAAD programs and design applications are not much usefull for designers with urbanistic design activities. Those applications can be utilized in design tasks, but they are not useful means to support the whole design process. To assist the urban designers in their design process, we need new CAD applications capable of providing comprehensive information to the users and supporting the urbanistic design process. To fulfil these requirments we have been working to develop an Urban-CAD program to overcome the limitations of the already existing CAD applications furthermore suit the urbanistic designers needs.

1. INTRODUCTION

A Lot of CAD programs and design applications exist to assist designers with their architectural and urbanistic design activities. Although those applications are useful means to be utilized in design tasks, they are not suitable tools for supporting the whole design process.

Using existing CAD programs (i.e. AutoCAD, Maya, 3D Studio Max and others) to create a design, the urbanist surely will encounter some limitations and basic problems of these programs. Besides the lack of a suitable data structure with specific capabilities, characteristics for processing and analyzing, there are three main limitations with regarding to the existing CAD programs.

First of all, all CAD programs are based on an architectural design process and therefore not suitable for the most urbastic designs. In the architectural design process the masses (walls of a building) are defined

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 387–399.
© 2001 *Kluwer Academic Publishers.*

according to the heartlines of the construction. The space is thus the result of the design process. This is also how the existing CAD programs work: defining a construction line and drawing a 3D 'solid', 'box', 'block', or 'mesh', regarding to the program. These objects define the space (see the following figure).

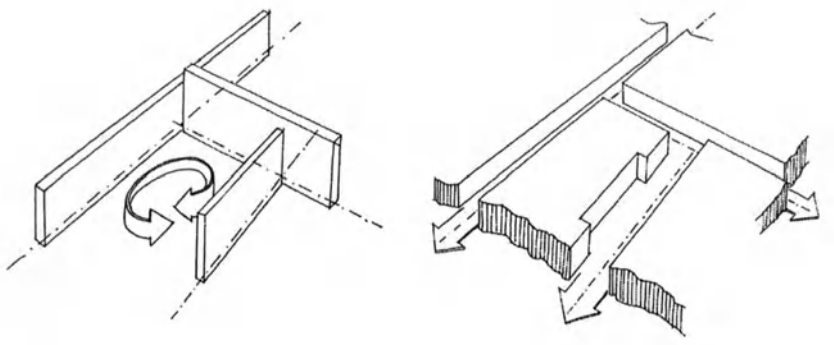


Figure 1. The architectural (left picture) and the urbanistic (right picture) design process

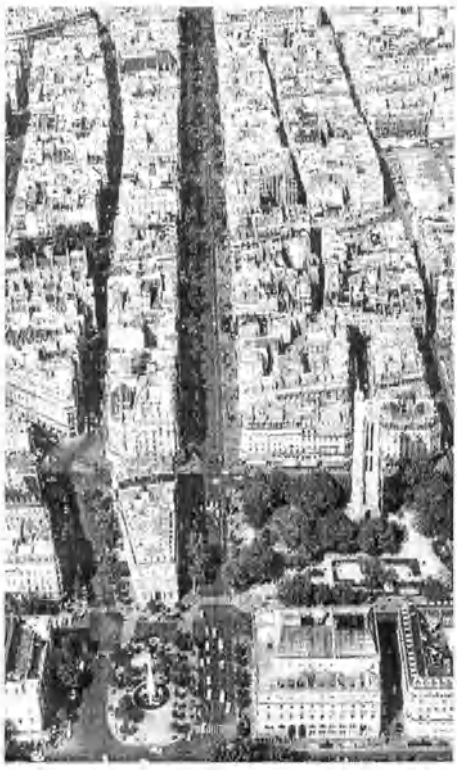
In an urbanistic design process, on the other hand, the designer starts defining the space according to the heartlines of the streets and leaving the masses (building blocks) as a result. In stead of creating a composition of masses, 'boxes' or 'solids', urbanism creates a composition of (public) spaces. The only exception is urbanism based on modernistic principles which in fact is a composition of masses and buildings. Example of the architectural and urbanistic design process are shown in figure 2.

Since this way of designing is not supported by any existing CAD program, this conceptual difference between architecture and urbanism creates the need for new CAD software based on the urbanistic design process.

Secondly, to achieve the whole design process, designers often must use two or more different programs. Programs that are excellent in 2D designing (i.e. AutoCAD but also CorelDraw and Canvas, which are based on vector drawings) mostly inappropriate in 3D designing or do not even provide that function. To visualize or render the drawing to get a realistic image, the design file has to be transferred to other programs, such as 3D Studio Max or Maya, which on their turn don't provide a sufficient way of 2D designing. If the designer wants to have the disposal of relevant data often a third program must be used.

In doing so, integration, interoperability, consistency of data, loss of time and other problems must be considered carefully. Apart from this there are also some tools for simulating an urban space (Breen, 1997), (Liggett,

Jepson, 1993 and 1995) but they also can not support the design process as well.



Above: example of architectural design process, the Barcelona pavilion of Mies van der Rohe.

Left: example of urbanictic design process, aerial view of Boulevard Sébastopol in Paris.

Figure 2. Different views of the design process

And thirdly, a design file mostly is a very detailed drawing, containing a lot of information. Often a drawing in any CAD program, therefore, will result in a tangle of lines, which have to be arranged in some 'layer structure', made by the designer. In order to show the relevant information layers have to be turned on or turned off.

Especially in urbanism a lot of different scales are covered and a lot of information has to be dealt with. Therefore an adequate way of displaying information is very important

Generally spoken, CAD programs do not create the possibility to experiment and design by using the computer and they are not capable of leading the designer through the whole design process. For urbanism, existing CAD applications are only tools to visualize an urbanistic design and to show the result of a design, but they are not a research and a design tool!

To overcome those problems we have been working on a research project to develop a CAD program that will assist the urbanistic designers with their design process.

Since any urban area consists of 3D objects like streets, buildings, and public and private spaces, by nature it is obvious that these objects should be designed in a 3D environment. Providing such an environment for the urban design community is the main reason behind this research project. Our Urban-CAD has been developed using an advanced CAD tool, MicroStation SE from Bentley Systems.

Two points were our main concerns in the development of Urban-CAD. First, the separation of the design elements and the information associated with those elements. Second, considering the different scales or abstraction levels that designers want to work at in the different design stages.

To achieve the first point, we have designed our own database management system that stores all associated information with any design object. The main objects in the application are: *streets*, *areas*, and *junctions*. Also a data structure has been used allowing the designers to manipulate the components of a main object as well as the sub-elements. Therefore, a street could have sub-elements such as a green-part, a pedestrian-part, and tram-lines. Also an area may contain parcels, building areas, and buildings. Having such a hierarchical structure of an urban environment makes our scaling mechanism a feasible task.

The scaling method, as our second concern, allows designers to work on their designs at various scales or abstraction levels. So in a certain scale the designer manipulates only those design objects that are associated with that scale. Working or viewing on a certain scale enables designers to focus only on an abstracted or a detailed level of an urban environment without being distracted by the related information in the levels above or below.

2. THE URBAN-CAD SYSTEM

An overview of the Urban-CAD system and the architecture of the system are given in this section.

2.1 Objects

The urban environment is composed of many small, medium and large objects. These individual objects are related to each other. They form combined objects which contain particular elements and details relevant to different scales. The main objects in the system are: *streets*, *junctions* (or intersections), *areas* (may include different parts such as a *green area*, a

public area, a *residential* or an *industrial area*, and other parts), *buildings*, and other objects such as *trees*, *lightings*, *bus* or *train stops*, and so on.

Note that the main objects of the system usually consist of several other things or parts. For instance, a street is not a single element, rather it consists of other parts, i.e. a pedestrian part, a part for bikes, a green part. These parts are usually parallel with the street. Also there exist other elements in a street such as trees (in the green part or even in the pedestrian part), lightings (may be in different parts). In Urban-CAD a street is drawn as a simple line in the design file. A profile, containing the different elements of the street will be defined, attached to the heartline, stored in database and displayed automatically on the screen. Note that in the existing CAD programs these parts must be drawn separately as well in the design file.

Also an area in our system is drawn as a simple shape, but information associated with it such as the type of an area, the scale it belongs to, its height, and so on are kept in the database. Therefore it is the responsibility of the system to figure out if an object has other parts or associated information to manipulate them while viewing or displaying the object on the screen.

2.2 Graphics environment

Any CAD system supports or provides a graphical environment for the users. Urban-CAD may have either its own graphical environment or based on one of the already existing CAD's graphics environment. We followed the second option and chose an advanced CAD tool, MicroStation SE from Bentley Systems. Therefore to work with Urban-CAD the MicroStation must have been installed on the user's computer.

Following the second way relieved us from developing a graphical environment from scratch. The user interface of Urban-CAD is the same user interface of the MicroStation SE, in addition with extra functions and utilities to support the urban design process. The user interface of the system is illustrated in figure 3.

As shown in the above figure some items are added to the menu items such as *Topography, Centerline, Area, Junction,* and *Urban-Tool*. Every added item includes specific functions and utilities associated with that item. For instance, to work with a *street* or an *area* object the user uses the *centerline* or the *area* menu item respectively. By clicking on any of these menu items, a small window will be opened up including other items with different functionalities. For example, the centerline menu item includes: *split line*, *merge line*, and *segment* functions.

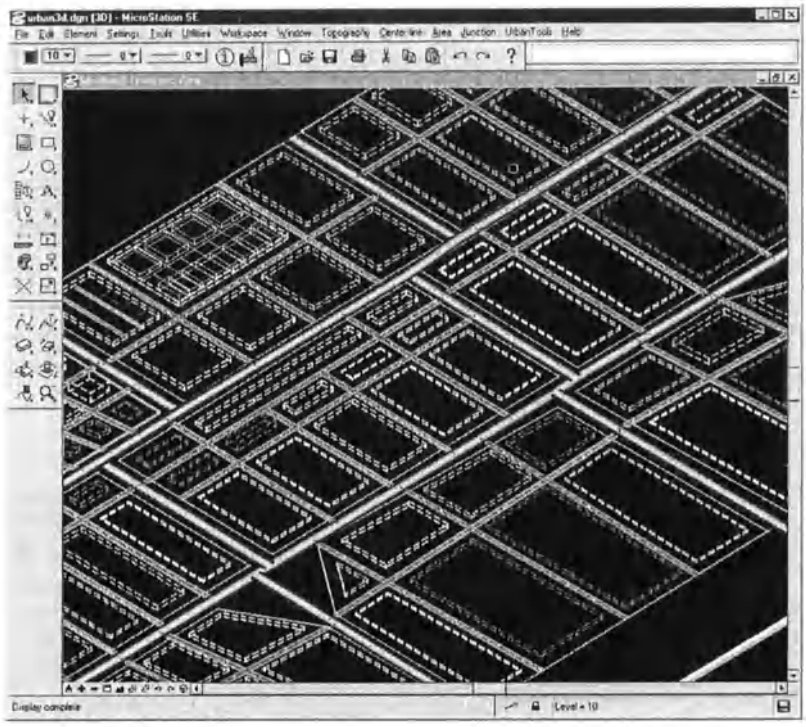


Figure 3. The user interface of Urban-CAD

2.3 Database management system

The Urban-CAD system has its own database for storing the related information to any urban object. Using a separate database management system provides a lot of flexibilities.

First, we can associate all information to an object as needed, so there is no need to attach such information as attributes of an object in the design file or draw it separately in that file. Separation of the related information with the object itself has many advantages. For instance, the design file of an urban space becomes so small. Also objects with their associated information, i.e. a street and its different parts, can be viewed or displayed differently at various scales. The scaling mechanism is explained in section three.

Second, an object, i.e. a street and its parts, in a design file is only shown or drawn by a single element such as a line. Therefore displaying or viewing of a street's parts will be done automatically on fly when the system displays that street object.

The structure of the Urban-CAD system

The following three components currently form the basis of the Urban-CAD software.

- Urban-CAD system provides a complete and integrated environment for designing or re-structureing an urban space.
- MicroStation SE Cad tool is a general-purpose computer-aided design and drafting package for 2D and 3D modeling and representation.
- Database management system is a general-purpose database for storing and organizing the associated information with all objects of the urban space.

These three systems provide the basic capabilities needed to plan and design an urban space. The overall structure of the Urban-CAD system is depicted in figure 4.

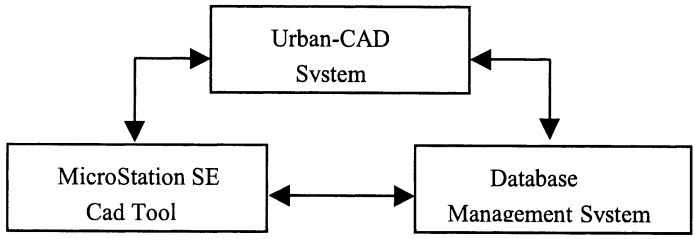


Figure 4. The structure of the Urban-CAD system

3. MAJOR FEATURES OF THE URBAN-CAD

Urban-CAD offers some new and interesting features to the users. The most important and useful features among the others are: the information storing and retrieving mechanism, the scaling method, displaying and viewing of the objects. Including these features in the system provide an integrated environment for urban designers to help in their design tasks.

3.1 Storing and retrieving information

The user interface of the Urban-CAD is shown in figure 3. This interface provides what users need for the urban design task. As mentioned before the user interface is the same as the MicroStation user interface with extra functionalities added to it. Clicking on the added menu items a small menu will be popped up showing the associated functions and utilities with that item. Any new function and utility does specific tasks. Figure 5 shows the

user interface with some dialog boxes representing the provided functions in them.

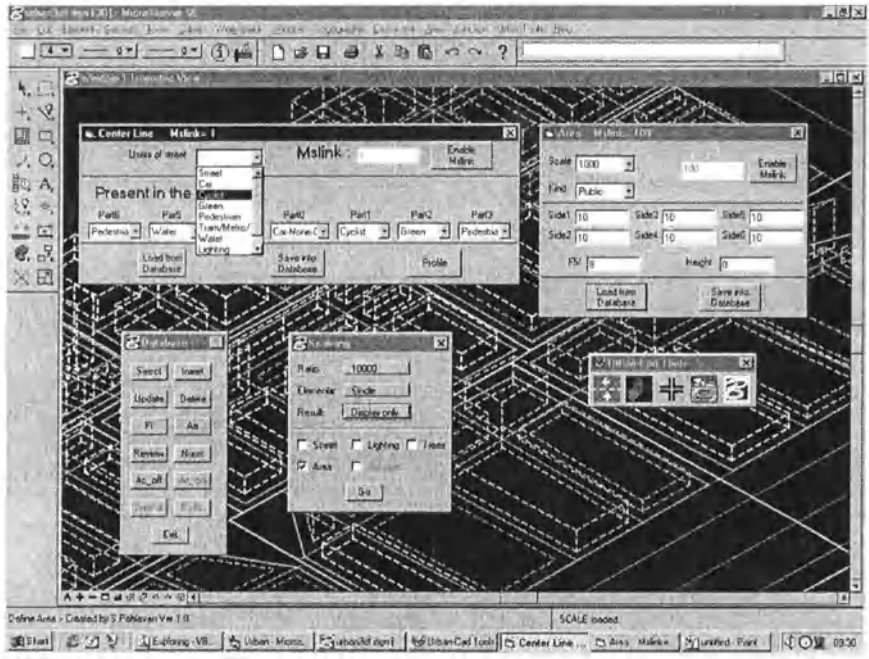


Figure 5. The user interface with some of the provided functions and utilities

Five dialog boxes are depicted in figure 5, the top two dialog boxes are used for storing and retrieving information of the street and the area objects. The bottom left dialog box is the main utility for interacting with the database management system. The bottom middle dialog box provides the scaling mechanism. The bottom right dialog box is the main box for invoking the provided functions and utilities of the Urban-CAD system. A short description of the dialog boxes illustarted in figure 5 is given here.

The *centerline* dialog box is for storing, retrieving, and manipulating the information of the street objects. As told before, a street may contain different parts (car, bike, green, pedestrian, tram/train, and water), each part can be used by any certain element. For instance, cars and pedestrians use the car and the pedestrian parts of a street respectively. These parts can be defined using the top left combo box of that dialog window. By clicking on any item of this combo box opens up another dialog box for defining that part. There exist some buttons in the centerline dialog box. Two buttons are used to store and to retrieve information from database. The profile button is for viewing the profile of a street. An example of a profile of a street is shown in figure 6.

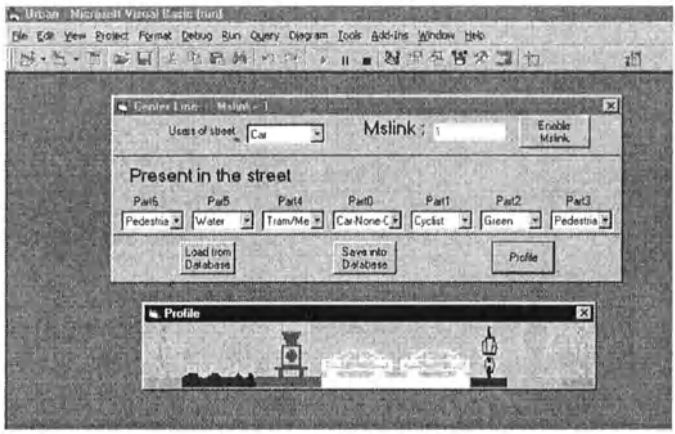


Figure 6. The profile of a street

As shown in above figure, by clicking on the profile button of the centerline dialog box the profile will be shown. As mentioned before, various parts of a street can be defined using the corresponding combo box to that part. In figure 4 these parts for a given street are: pedestrian, water, tram/train/, main part (car-none-car), cyclist, green, and pedestrian. The important point in displaying the profile is the size of those parts. The size is displayed dynamically, meaning that depending to the width of a part of a street, this part is shown proportionally to the total size of the profile. As the profile shows the main part of the street, the cars part, has the biggest size. The other parts occupy a portion of the profile with respect to their given width. The profile helps very much the urban designers to make decision while they allocate spaces to a street and its parts

The information of an area object is manipulated using the *area* dialog box. The *scale* (is discussed in the next section) and the *type* (may be green, public, building, or mix of these) fields are defined through this dialog box. Storing information to and retrieving them from database are done using the certain button of the dialog box.

The *scaling* dialog box (depicted in figure 5) allows displaying objects of an urban space in certain scale. The *ratio* button allows choosing one of the six defined scales. The *elements* button lets the user to select one of the three (single, fence, or file) modes for displaying. In the single mode only one object can be displayed. Selecting the fence mode, all selected objects inside a drawn fence are displayed. The file mode allows displaying all objects in the design file. For now four option buttons are provided in the bottom of this dialog box. They are: *street*, *lighting*, *tree*, and *area*. Checking these options does certain things. For instance, if the user wants to see or view

only the street or the area objects in the file he must check the corresponding option. Selecting both options allows displaying the street and the area objects. Also if users check the lighting or the tree option (or both) the Urban-CAD system displays those objects (trees and lighting) of a street if exist while displaying that street. After selecting the options users must first click on the *go* button and then they click on any place of the design file to let the system do the job.

The *database* dialog box provides many useful functions and utilities that users need to interact with the database management system. In other words this dialog box provides means of interaction between the users and the database.

3.2 Scaling mechanism

Scale is a very important aspect in our Urban-CAD program. In principal, designing by computer is without any specific scale. A design-file therefore will be as detailed as the user will detail it. Often this will result in a very detailed design-file and a computer screen with information overkill. Mostly the designer has to turn off some 'layers' to reduce the amount of information and the level of detail in order to keep a general overview. This work is confusing, taking a lot of time and slowing down the design process.

Since urbanism focuses on a lot of different scales, from defining the pedestrian pavement to making plans on a national and even international level, designers often work on different scales at the same time. Aware or unaware a lot of decisions are therefore made at a wrong scale and a wrong moment in the design process. One of the main targets in Urban-CAD, therefore, is to show the designer the right information at the right moment in the design process.

Whether an element of the design-file is displayed on the computer screen, depends on the scale the designer is working at. For example a street is shown as a single line when working on the scale of the whole city. On the scale of a neighborhood this same street will be shown as a strip with a certain width, containing different parts (i.e. car part, bike paths and sidewalks). Six different scales of observation have been considered, they are: 1:10000, 1:3000, 1:1000, 1:300, 1:100, and 1:30.

The predefined scales that the design elements belong to, are based on the research done by prof.dr.ir. T.M. de Jong of the faculty of Architecture of Delft university of technology. In his book (De Jong, 1995) the relation between scale and the perception of information has been explained. It shows the mathematical relation between a specific scale and the objects seen at that scale.

Using this theory in our scaling system, Urban-CAD automatically displays the relevant information belonging to the selected scale.

3.2.1 Scaling the street objects

As described before, a street object must belong to one of the six predefined scales ranging from 1:10000 to 1:30. As explained in section two, users can view or display an urban space using the scaling dialog box. Through this window users select a certain scale and choose options for viewing. Either the street objects or the area objects (also both objects) can be selected. For example, when the system displays at the scale 1:3000, all street objects that belong to this scale and the scales above it (in this case only the scale 1:10000) will be displayed.

It has been explained that a street object may consist of various parts. Depending on the width of these parts they may be shown as well at the displaying scale. If the width of a part is bigger than a certain value, that part will be displayed at that scale. For instance, at scales 1:10000, 1:3000 and 1:1000 if the width of a part is bigger than 50, 20 or 5 meters respectively that part will be displayed at the corresponding scale. Note that, if a part must not be displayed with respect to this rule, the width of that part always is included in the total width of the street.

The deployed scaling mechanism allows the users to focus only on the objects that are relevant to a certain design stage. While they work at high scales, 1:10000 or 1:3000, they concern on boulevards and big roads. So the system views or displays only this type of street objects. But while the users are working at middle scales, 1:1000 or 1:300, they can view or display the normal size street objects. When they work at the lowest scale, 1:30, they can even view the sidewalks and the design of the public space. Figure 7 shows some street objects of an urban space in the lowest scale.

3.2.2 Scaling the area objects

After defining or drawing the heartline of the street objects, the area objects can be defined automatically. The space between three or more street objects is seen as an area object. In the design process there are some steps to be taken to develop the area objects and the future buildings within that area. The steps are: defining the areas, defining the FSI (Floor Space Index), defining the building-line, parceling the area, defining the building envelop, and defining the buildings. The FSI is a value representing the relation

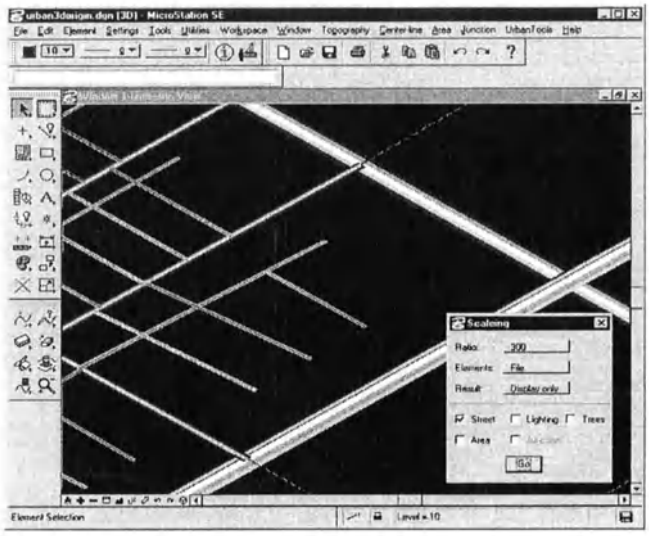


Figure 7. Some street objects are displayed in the scale 1:30.

between the area and the square meters floor space that has to be realized, in other words it represents the *density* of the area. Knowing the FSI of an area, we can compute the height of that area. The height of the areas will be used by the system to display those objects in 3D representation mode.

As illustrated in the user interface of the Urban-CAD system, the menu item called *area* contains functions and utilities that a user needs to do the aforementioned steps. Some steps will be done automatically when the user gives the needed information. Other steps will be achieved with little effort and using the provided utilities of the system.

Any area object must belong to one of the four scales ranging from 1:10000 to 1:300. Also an area object may consist of some smaller sub-areas. In this case, each sub-area must belong to a lower scale than the main area object belongs to. When the user views or displays objects in a certain scale, only the area objects that belong to this scale will be displayed. On the contrary, the street objects belonging to the higher scales will be displayed as well.

4. CONCLUSION

Due to the limitations of the existing CAD programs and the inadequacy of those programs to assist designers in urbanistic design activities, we have been working on a research project to develop a CAD program to overcome

those limitations and also suit the needs of the urbanistic design process. This research work is a project of the OSF (Onderwijs Stimulerings Fonds) of Delft university of technology. Professor J. Heeling from the faculty of Architecture who initiated this project (Heeling, Meyer, et al, 1997). The basic goal was the development of a computer application in order to make the urbanistic design process by computer easier, quicker and more effective. It is expected that students will start working with a prototype of Urban-CAD software in October 2001.

The well known and advanced CAD tool, MicroStation from Bentley Systems, has been used as the graphical environment of the Urban-CAD system. Therefore the system has been developed on top of this CAD tool. The scaling mechanism of the system is the outstanding feature that is missing almost in all the existing similar applications. This feature plus the others makes Urban-CAD an advanced application that the urbanistic designers community needs.

5. REFERENCES

Breen, J., and J. van der Does, 1997, "Architectural and urban simulation techniques in research and education", in: Third Conference of the European Architectural Endoscopy Association, August 27-29, Delft.

De Jong, Taeke M., 1995, "Culturele Variatie", in: technical report of Technische Ecologie & Milieuplanning, Technische Universiteit Delft, Faculteit Bouwkunde.

Heeling, J., V. Meyer, J. Westrik, and et al, 1997, "The basics of Urbanism in the 21st century", in: (technical report) Blauwe Kamer, Delft, Architecture Department.

Liggett, R and W. Jepson, 1993, "An integrated environment for urban simulation", in: Proceedings of the Third International Conference on Computers in Urban Planning and Management '93, Atlanta, p. 565-583.

Liggett, R. and W. Jepson, 1995, "Implementing and integrated environment for urban simulation", in: Koutamanis (ed.) Visual Data Bases.

Unfocused Interaction in Distributed Workgroups

Establishing group presence in a web-based environment.

Brian R. Johnson
Design Machine Group, University of Washington

Key words: Distributed workgroups, unfocused interaction, presence, collaboration

Abstract: Face-to-face human interaction is divided into "focused" and "unfocused" types. Unfocused interaction often conveys important content and context information and contributes to group cohesiveness and effectiveness. Research in Computer Mediated Communication (CMC) and Computer Supported Collaborative Work (CSCW) is also concerned with human interaction. CMC tools, such as electronic mail, and CSCW tools, such as Decision Support Systems (DSS) and Group Support Systems (GSS) provide for focused interaction among members of distributed workgroups. However, little has been published regarding unfocused interaction in distributed workgroups, where group members' primary work activities hold "center-stage" and communication activities are peripheral, though this describes many distributed educational and work situations. A framework for studying this type of support using standard web browsers and server applications is described, and informal preliminary results are discussed. Opportunities for future support of peripheral awareness and unfocused interaction are also discussed.

1. INTRODUCTION

Architecture has always been concerned with physical space and the ways in which people use that space to work, learn and play together. Recent advances in computer networks have made it possible for people to conduct much of their lives online, and given rise to terms like "cyberspace," "media space" and "virtual space." Through telecommuting, distance education and the like, these metaphorical spaces promise to alter how we build and inhabit physical space, but perhaps the sensitivities and experiences of those who study the human use of environments would benefit the new electronic

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 401–414.
© 2001 *Kluwer Academic Publishers.*

worlds as well. One particular area of consideration involves "unfocused interaction".

1.1 What is Unfocused Interaction?

During the 20th century sociologists, anthropologists and others have studied the interaction of individuals in common social and work settings, including task-oriented work groups, semi-formal groups such as classrooms, and casual assemblies of individuals in public places. In his 1963 book *Behavior in Public Places* Erving Goffman characterizes interaction along a number of axes. One axis involves the distinction between "unfocused interaction ... the kind of communication that occurs when one gleans information about another person by glancing at him", and "focused interaction, the kind of interaction that occurs when persons gather close together and openly cooperate to sustain a single focus of attention, typically by taking turns at talking." (Goffman, 1963, p24).

1.2 Importance of Unfocused Interaction

Thus, face-to-face human interaction can be divided into "focused" and "unfocused" types. Unfocused interaction, also referred to as *peripheral awareness*, *passive awareness*, and *peripheral monitoring*, is unintentional but often conveys important content and context information and contributes to group cohesiveness and effectiveness. One classic example involves two London railway workers and the information passed passively between them simply because they could overhear each other's phone conversations (Heath and Luff, 1996).

The education of designers has leaned heavily on unfocused interaction. In *Educating the Reflective Practitioner*, Donald Schön argues that the passive forms of communication are critical to the socialization process which is part of educating designers (and, by extension, performing design). While an individual "desk crit" is largely a focused interaction, he and others also see significant value in the unfocused interactions, involving overhearing, student-to-student interaction, and spontaneous discussion. This is why schools of architecture utilize the costly studio process, and identify "studio culture" as one of their valuable assets. In an era where more and more emphasis is shifting to on-line education (Carnevale, 2000) it is important to understand which aspects of our educational process might be compromised or supported by deployment of these technologies.

1.3 Impact of Computers and Networks

With the development of computer networks in the 1970s, programmers began exploring ways of using them for human-human interaction, giving rise to the field of Computer Mediated Communication (CMC). Many CMC applications, such as electronic mail, provide generic communication tools which might be applied in a variety of situations. Other research focused on supporting shared tasks over distance, spawning the field of Computer Supported Collaborative work (CSCW). Within CSCW decision-making in distributed work group situations has been studied using Decision Support Systems (DSS) or Group Support Systems (GSS).

1.3.1 Focused Interaction

This area of computing has tended to focus on the technical aspects of communication, enabling more and more modes of exchange, from text messages to spoken words, to video images. Most of the software created to date addresses the need for direct, person to person(s) communication. These systems share an orientation to Goffman's "focused interaction". They fulfil an interpersonal communication need when consciously sought by the participants. They are not generally intended to facilitate the indirect or passive forms of communication described as "peripheral monitoring", eavesdropping, and so on.

The common interaction tools available today provide most of the conceivable modes of focused interaction (Johnson, 2000a). Their availability has also demonstrated that CMC creates a social interaction space (Turkle, 1995). Research by Turkle and others has looked at the impact of CMC tools on individual behaviours. This attention has focused on behaviours in the "public" places of the Internet, by looking at how people use Multi-User Dungeons (or Domains) (MUDs), Internet Relay Chat (IRC), and email in their social, discretionary lives. It has not examined as closely the use of these tools in professional practice or education domains.

1.3.2 Unfocused Interaction

We know individuals interact in unfocused ways. However, "there is little research on what role passive awareness itself plays in group work activity and cohesion" (Dourish and Bly, 1992). When available, formal evaluations of GSS tend to ignore the long-term team-building aspects of the systems (George and Jessup, 1997). Nonetheless research has moved forward on the intuition that these interactions are important. Projects which have incorporated support for unfocused interaction include the *Portholes*

project at Xerox (Dourish and Bly, 1992), *Beyond Being There* (Hollan and Stornetta, 1992), and recent work demonstrating higher rates of user satisfaction when using a task-oriented CSCW system incorporating presence elements (Gutwin and Greenberg, 2000). Researchers also are exploring the ways in which peripheral awareness matters in technologically mediated communications (Monk and Watts, 2000).

1.3.3 Benefits of Passive Awareness in Shared tasks.

As indicated above, researchers have begun to look at the use of CMC to enhance and support users accomplishing shared tasks in Computer Supported Collaborative Work (CSCW) and have demonstrated that appropriate "workspace awareness" tools can, indeed, enhance worker satisfaction and productivity in focused tasks (Gutwin & Greenberg, 1999).

1.3.4 Benefits of Passive Awareness in Social Interaction

Examples of benefits deriving from a general passive awareness are more anecdotal. Turkle recounts the story of a tele-commuting software tester who was only able to work well once they established a habit of opening a chat window with a colleague (Turkle, 1995).

2. THE COMPADRES FRAMEWORK

The Compadres system was developed to investigate the hypothesis that support of unfocused interaction in open-ended tasks, such as students and professionals completing collaborative design and course work, is beneficial, and may be critical to the success of human interaction involved in educational and design processes.

Of particular interest is the support of loosely-coupled distributed workgroups. This encompasses not only full time workers, but also mutual support in seminar courses, on-line office hours for educators, distance education groups, and research groups. No particular agenda was presumed for the group, except some decision to "be" a group for a period of time. That is, the system is not task-oriented. It is oriented to improving communication and cohesion within the group.

We had prior academic experience with distributed work groups using a variety of both real-time and asynchronous tools (Kolarevic, Schmidt, *et al.*, 1998) (Donath, Kruijff, *et al.*, 1999). That experience indicated that support for both asynchronous and synchronous communications was important, as was simplification of the communication options.

2.1 Web-based

The idea of passively linking distant locales is not new. In addition to pre-web projects like *Portholes* (Dourish and Bly, 1992) and *Beyond Being There* (Hollan and Stornetta, 1992), the web-based *Phase-(x)* system (Wenz and Hirschberg, 1997) demonstrates varying support for the idea.

Compadres combines an extremely light-weight client presence, a focus on communication, and a visual presence monitor.

The system uses a web browser as the client software, with several web server Common Gateway Interface (CGI) applications and databases to store system data. As in the Portholes system, existing CMC tools (email and chat) are used for focused communication.

2.2 Lightweight Client Interface

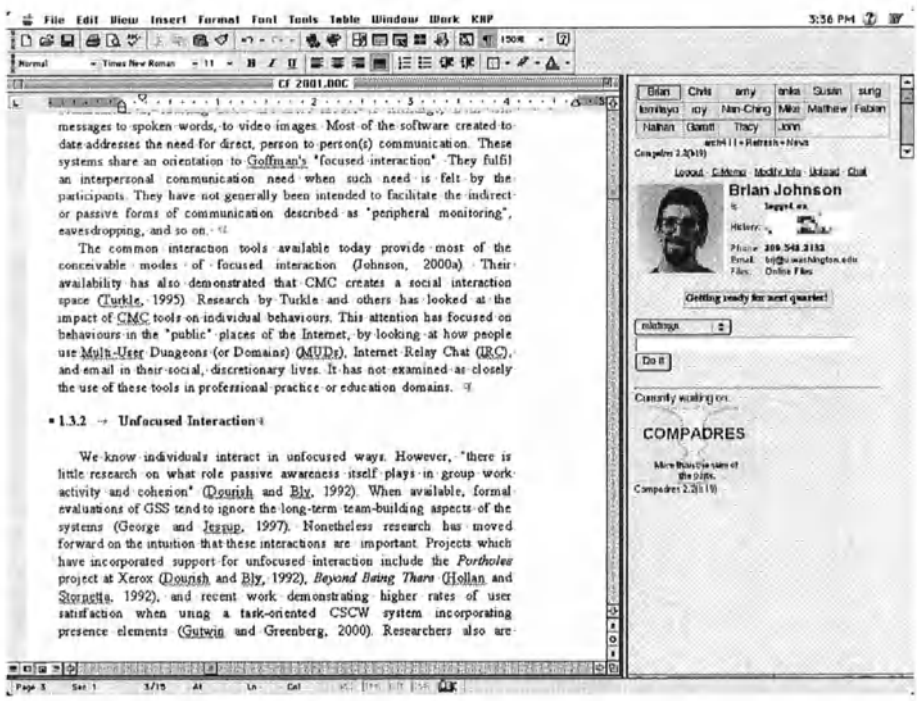


Figure 1. Compadres in use along-side primary-task application window

The current client interface (shown to the right in Figure 1) uses a web browser as the host environment. This choice supports itinerant users (such as students) because web browsers are practically ubiquitous and users' personal data and preferences are stored in the central Compadres database. They can sit down at any networked workstation with a web browser and

access Compadres by simply directing the browser to the correct URL. Browser features are generally familiar to users, reducing cognitive load associated with using the system.

The Compadres browser window is purposely narrow, so that it can be located to one side of a user's monitor. It is divided into three frames. The top frame houses the presence monitor. Below this is a command console, and below this is space in which user data pages and transient interaction pages are presented.

2.2.1 The Presence Monitor

While it is web-based, Compadres is a private system, limiting access to known users. The use of usernames and passwords may make users feel more comfortable with personal information and pictures being on the web, but it also means they know who the other users are. This has been shown to encourage use (Nunamaker, 1997). It also allows the system to actively tracks users. One product of this tracking is the real-time presence monitor.

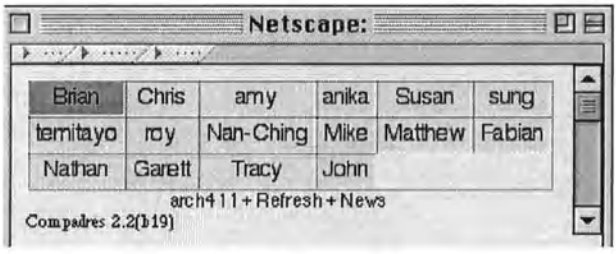


Figure 2. The Compadres presence monitor, showing connected users in green (dark grey) and absent ones in pink (light gray)

The presence monitor consists of a text-based area of the screen which simultaneously shows current workgroup ("cadre") membership and individual connection status (through varied background colors–green for connected users, pink for absent ones. This page automatically refreshes two or three times a minute.

Each time the presence monitor is refreshed the CGI queries the user database for new messages. If present, it delivers them in a JavaScript fragment that issues an alert upon loading, assuring that they will be seen even if the Compadres window is covered.

Links at the bottom of the presence monitor (see Figure 3) allow the user to see the name of the current cadre, select an alternative group, and manually refresh the presence monitor, and view system-wide news

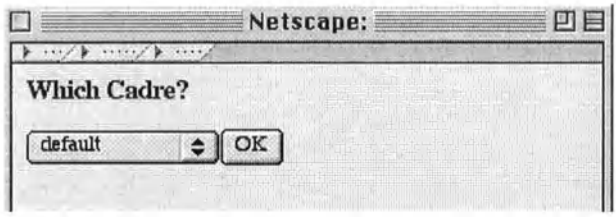


Figure 3. Cadré selection pop-up displayed when changing workgroups

2.2.2 The Command Console

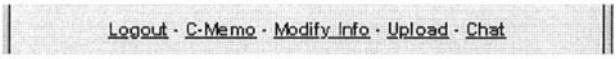


Figure 4. Command console, showing the user's perpetual options

The thin central frame of the interface (Figure 4) provides the user with certain options that are always available, as described in Table 1.

Table 1. Command console options

Command	Action
Logout	Terminate Compadres services.
C-Memo	Compose and send a message to multiple users selected by checklist from the Cadre's membership.
Modify Info	Display your own user data for modification.
Upload	Link to URL for manipulation of group files.
Chat	Link to URL for group chat-room.

2.2.3 The Personal Data Page

Each user of Compadres has a personal data page. This page, as shown in Figure 5, displays the individual's "public" face, the information that they wish to share with other workgroup members.

Another product of the system's awareness of users is maintenance of a "radar view" of presence over time (see "History" field in Figure 5). A grey-scale density map showing the relative amount of time the user was connected over the previous seven days is presented as part of their personal data page. The green (dark) squares along the edges indicate the current hour and day of the week.

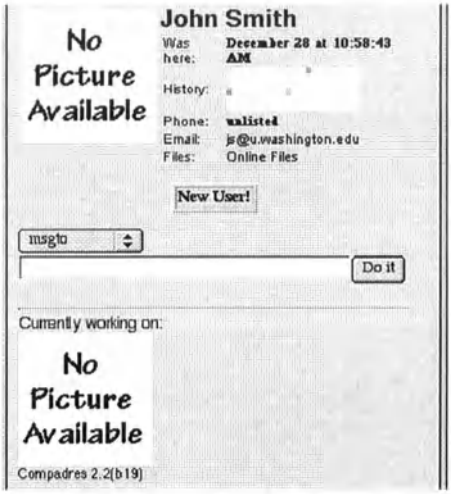


Figure 5. Personal Data Page, showing default values for a new user

User data pages present the following personalizable options:

Table 2. Individual user-alterable preference fields

Compadres User Preferences
Head-shot (IMG) URL
Head-shot destination (link) URL
Name (permitting use of nick-names, etc.)
Phone number
E-mail address
Personal files URL
"Door Sign" text (shown against a yellow background)
Personal HTML (arbitrary)
Current work (IMG) URL
Current work destination (link) URL
Cadre subscription list (listing cadres to which the user belongs—not displayed in Figure 5)
Default cadre (establishes initial presence monitor—not displayed in Figure 5)

Users may change their preferences through selection of the "Modify Info" command console option.

When viewing the data page of another user, you may use the Message-to form to write them a quick note. When viewing your own data page, this same input field is used to quickly update the Door-sign message. Messages are queued for the recipient in the user database and delivered the next time they log on or their presence monitor refreshes. Uncollected messages are forwarded via e-mail each night, when possible.

2.3 Server Side Structure

The Compadres system consists of a standard (Macintosh) web server, some HTML files, the Compadres (Applescript) CGI applications, and a (FileMaker Pro) database of user and system data. Individual user images and files may exist anywhere on the web, only links are stored in the Compadres database.

The Compadres CGI provides users with a private environment. Each request on the CGI must be authorized. Since it would be impractical to include a username and password with each hit, Compadres uses a trust model. When a user logs-on to the server using their Compadres username and password, the CGI records their IP-address. Subsequent requests from that IP-address are trusted until the user logs-out or times-out. Timing-out covers contingencies such as users who close the browser window without logging out. Further, since each interaction with the CGI, including updating the presence monitor, resets the time-out clock, "timing out" and "logging out" are virtually synonymous.

The Compadres application processes all requests on the database, displaying and updating user data as needed. It also logs connectivity data to the individual history fields. Once a day it launches a utility to convert the history data into the history graphics displayed as part of the user's personal data page, forward uncollected e-mail, etc.

2.4 Alternative Client Implementations

In the real-world, unfocused interaction takes on many forms, including acoustic, visual, and even olfactory ("Jan's brewing coffee again"). We do not need to "log on" to take advantage of these cues. In our ideal system, users would not need to explicitly "connect" or "disconnect". Nor would screen space be required for the presence monitor, cues might be embedded in the surrounding physical environment.

One configuration of the Compadres system involves running two special programs on the client's machine in addition to a browser. One provides an alternative to a static "current work" desktop graphic by broadcasting live images of the user's screen buffer (suitably downsized to remove detail). The other replaces the "head shot" graphic with a live webcam image.

Other configurations eliminate the browser window. In one we developed a "Butler" client for Compadres. When launched, this client-side script retrieves the presence-monitor data, scans it for newly-connected or disconnected users, and then speaks an announcement to the effect that the individual has arrived or departed.

In a related "soft media" project, the Butler was made to activate "Crickets", generating unique user-specific tone sequences at intervals whenever monitored individuals were on-line. A variation of this, using X-10 technology and lamps, is also under development.

2.5 Initial Results

Researchers in CSCW have previously noted that evaluation of passive awareness systems used by groups is difficult because direct attention interferes with their unfocused nature (Dourish and Bly, 1992) and because the complexity of the surrounding "social, motivational, economic, and political dynamics" (Grudin, 1994) may mask results.

Compadres has been used experimentally as part of three courses. In each case, students were encouraged to use the system, but were not required to, and had ready access to face-to-face communications. After using the system for a few days, they were asked to provide feedback. Table 3 shows some of the comments received. Initial student response was also reported in (Johnson, 2000b).

Table 3. Comments from several student users of Compadres

#1	"Works as well as any other instant messaging service" "A great feature is having the person's picture and bio on the page" "It would have been nice to have a place to archive messages, " "I [found] it annoying that any messages that [were] sent to you offline pop up instantly as you log in"
#2	"I think Compadres works very well "
#3	"Compadres ... is a wonderful communication tool. I believe the two things I most appreciated were including the picture of the person you are talking to and the 'connected/disconnected' green buttons." "You could enable a 'history of messages received'"
#4	"I knew then that I was part of some group" "Most of our discussions ... occurred face to face"
#5	"Compadres was not used [extensively] because we could get up and talk to each other and see what our peers are working on." "My family is spread all over the U.S. ... I think it would be interesting to bring us together (virtually) [in] ... a network situation such as Compadres"
#6	"We didn't use Compadres ... much because we basically knew where people were all of the time." "I could see where members of a team ... could really rely on [it] for basic communication."

These responses reflect the challenge of constructing suitable "distributed workgroup" situations in a residential university, given the availability of face-to-face alternatives. Nonetheless, these preliminary results suggest that Compadres does contribute to group identity and cohesion, and does present users with recognisable benefits. Even the preference for face-to-face interaction can be seen as an indicator of the value and importance of workspace awareness, whether focused or unfocused.

2.5.1 Personalization

The personal data page, including a photograph, was appreciated by users for it's group-building, though some users replaced the photo with a different graphic. Very few students provided a phone number (though cell-phone use is quite wide-spread in this group).

2.5.2 Messaging

The messaging feature drew significant use and comment. Though the "message to" input field is only 40 characters wide, it was common to find substantial messages being transmitted with this mechanism. Several responses indicated desire for more options in message management. This has been partially addressed through addition of the "Compadre Memo" command-console option, and the nightly forwarding of unread messages via e-mail.

2.5.3 Privacy

Management of personal privacy is certainly an issue, not unique to Compadres. The Portholes Public client, for example, which was presumably available to users outside the primary user group, didn't show office spaces (Dourish and Bly, 1992, p544). Others have identified privacy concerns as critical to the acceptance of CMC (Tu, 2000). Many objections expressed relative to web-cam software involve the anonymity of possible viewers. Careful integration of such technology into an overall privacy management scheme seems essential.

3. FUTURE DEVELOPMENT

While more than two years old, Compadres is still early in its development. There are a number of areas undergoing active development, and one or two in which we hope to undertake projects.

3.1 Increased Passivity

One of the motivations for creating Compadres was the observation that the available tools require considerable user knowledge and commitment to use. Increasing ease of use of the tools is one of the important CMC agenda items (Nunamaker, 1997). We felt that one source of difficulty was the number of different interfaces and attempted to mitigate this complexity through use of a single "switchboard". Currently Compadres brings together email and chat within a single umbrella.

The motivation behind Compadres is to link people, not places. Thus, it is necessary to identify individual participants. Unfortunately, biometric and similar technology is of little use unless widely deployed. However, several scenarios exist in which a single machine is consistently used by a single user. We are examining opportunities to provide such users with automatic log-on and log-off, by monitoring system mouse and keyboard events for example.

Increased passivity through alternative presence monitors using audio or alternative visual cues (as described above) is also an area for investigation. Alternative mechanisms for displaying and managing availability cues (perhaps tied to physical objects in the office) offer additional avenues.

3.2 Increased Interaction

Another area of investigation is that of mechanisms for making available the kind of information available in physical environments. In the real world, we can see someone looking in our office door. We can see who is visiting whom in our lab. Not only can we see, we are able to eavesdrop if their conversation is loud enough (and receive tacit permission to do so by the speaking volume used by the participants). Replicating such interactions, especially asynchronously, presents a number of visualization, presentation, and privacy-management challenges.

3.3 Measuring Benefits

As indicated above, assessing the impact of CMC systems such as this is difficult. We are approaching this from two sides: we are working with small

groups to asses responses to the system, and we have instrumented the system to log user actions, providing objective background data. We are also seeking appropriate deployment settings in which to conduct more extensive testing.

3.4 Integration into other tools

Unfocused interaction in the physical world is a side-effect of our bodily embedding in that world. The natural domain for embedding unfocused interaction tools in the CMC universe is the operating system, where keyboard and mouse events provide information regarding the user's presence, and might be indicative of more subtle mental states (e.g., concentration).

At the same time, inclusion of "unfocused interaction" features in "primary task" CSCW applications (such as multi-user CAD software for architects and engineers) might significantly enhance collaboration even within co-located workgroups.

This is a direction we are beginning to explore.

4. CONCLUSIONS

Face-to-face communication between people clearly remains the vastly preferred communication option when such interaction is available (Hollan and Stornetta, 1992). However, as CMC matures, focused interaction tools such as video conferencing and email will be joined by tools which support unfocused communication. These tools will help maintain workgroup identity and group awareness for those who work at a distance.

It is important to continue the investigation of feature-sets, interaction modes, and user behaviors that accompany these tools. The Compadres framework provides several opportunities to advance this research.

5. AVAILABILITY

We believe that one of the best ways to understand collaborative work systems is through collaboration. Individuals or groups wishing to try Compadres to support a workgroup, or wishing to collaborate on development of the system, should contact the author at brj@u.washington.edu, or visit the Compadres web site at <http://www.caup.washington.edu/software/compadres>.

REFERENCES

Bradford, J.W., N. Cheng, and T. Kvan. "Virtual Design Studios", in *Proceedings of the 12th eCAADe conference*, pp 163-167.

Carnevale, D., 2000, "New Master Plan in Washington State Calls for More Online Instruction" in *The Chronicle of Higher Education*, (46) #22, February, 2000, p A50.

Donath, D. E. Kruijff, H. Regenbrecht, U. Hirshberg, B. Johnson, B. Kolarevic, J. Wojtowicz, "Virtual Design Studio 1998 – a Place2Wait" in A. Brown, M. Knight, P. Berridge (eds.) *Proceedings of eCAADe 17 – Architectural Computing from Turing to 2000*, pp 453-458.

Dourish, P. and S. Bly, 1992, "Portholes: Supporting Awareness in a Distributed Work Group", in *Proceedings of SIG CHI '92*, pp 541-547.

Goffman, Erving, 1963, *Behavior in Public Places,* The Free Press of Glencoe.

Grudin, J., 1994, "Groupware and Social Dynamics: Eight challenges for developers", in *Communications of the ACM*, (37)1, pp 93-105.

Gutwin, C. and S. Greenberg, 1999, "The Effects of Workgroup Awareness Support on the Usability of Real-Time Distributed Groupware" in *ACM Transactions on Computer-Human Interaction*, v 6, # 3, pp 243-281.

Heath, C. and Luff, P., 1996, "Convergent Activities: line control and passenger information on the London underground" in Y. Engestrom & D. Middleton (eds.) *Cognition and Communication at Work*, pp96-129, Cambridge: Cambridge University Press.

Johnson, B. R., 2000a, "Sustaining Studio Culture: How Well do Internet Tools Meet the Needs of Virtual Design Studios?" in D. Donath, (ed.) *Proceedings of eCAADe 2000, Promise and Reality*, pp 15-21.

Johnson, B. R., 2000b, "Between Friends: Support of Workgroup Communications" in M. Clayton and G. Vasquez de Velasco, (eds.), *Proceedings of ACADIA 2000, Eternity Infinity and Virtuality in Architecture*, in press.

Kolarevic, B., G. Schmidt, U. Hirschberg, D. Kurman, & B. Johnson, 1998, "An Experiment in Design Collaboration" in T. Seebohm and S. van Wyk, (eds.) *Proceedings of ACADIA '98, Digital Design Studios: Do Computers Make a Difference?*, pp 90-99.

McCall, R., S. Holmes,. J. Voeller, & E. Johnson. "World Wide Presentation and Critique of Design Proposals with Web-PHIDIAS" in T. Seebohm and S. van Wyk, (eds.) *Proceedings of ACADIA '98, Digital Design Studios: Do Computers Make a Difference?*, pp 254-265.

Monk, A. and L. Watts, 2000, "Peripheral participation in video-mediated communication", *International Journal of Human-Computer Studies*, 52, p. 933-958.

Nunamaker, J. F., 1997, "Future research in group support systems: needs, some questions and possible directions" in *International Journal of Human-Computer Studies*, 47, p 357-385.

Schön, D. A., 1987, *Educating the Reflective Practitioner*, San Francisco: Jossey-Bass.

Tu, C. H., 2000, "Critical examination of factors affecting interaction on CMC" in *Journal of Network and Computer Applications*, 23, p 39-58.

Turkle, S., 1995, *Life on the Screen: identity in the age of the Internet.* New York: Simon & Shuster.

Vasquez de Velasco, G. and D. Hutchison, 1999, "An Immersive Environment for Virtual Design Studios" in O. Ataman and J. Bermudez, (eds.), *Proceedings of ACADIA '99, Media and Design Process*, pp 328-329.

Wenz, F. and U. Hirschberg, 1997, "Phase(x) Memetic Engineering for Architecture", in B. Martens, H. Linzer, and A. Voigt (eds.), *Proceedings of the 15th eCAADe Conference Challenges of the Future.*

Wojtowicz, J. (ed.), 1994, *Virtual Design Studio*, Hong Kong: Hong Kong University Press.

Coordination of Distributed Design Activities: A Rule-Driven Approach

Taysheng Jeng
Department of Architeture
National Cheng-Kung University, Taiwan

Key words: Coordination, Dependency, Design Process Models, Collaborative Design

Abstract: The purpose of this work is to develop effective, yet flexible, methods of coordination in support of collaborative design. This work takes a rule-driven approach to modeling process knowledge for coordination of distributed activities. In this paper, coordination knowledge is represented by a set of dependency rules that determines when, how, and by whom each activity is performed. A design coordination model called DCM is introduced in terms of the dependency rules and the activity states associated with the use of the whole range of computer applications. DCM permits coordination flexibility and allows visibility of coordination logic dealing with dynamic aspects of design processes. A web-based system prototype that implements the design coordination model is demonstrated. Opportunities for using the system prototype to aid design coordination are discussed.

1. INTRODUCTION

Advances in computer networks and digital communication have provided new opportunities for multiple designers to collaborate from different positions in time-space. While new technologies have enabled remote collaboration, coordination remains a challenging problem in most large-scale multi-organization collaborative design projects. An initial recognition is coordination knowledge is a basic aspect of professional expertise that is currently carried in the heads of consultants or the senior people in the architectural firm. There has been no way to represent tacit coordination knowledge or measure it (Eastman et al. 1999).

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 415–426.
© 2001 *Kluwer Academic Publishers.*

Design coordination can be facilitated but not automated. Design processes often involve multilevel collaborative tasks and inter-dependent activities. Most management processes in designing may reach a particular level where autonomous coordination must take over for managing dependencies between activities. In the level of autonomous coordination, the process to coordinate the work cannot be pre-defined. The structure of the inter-dependencies between design activities usually is also ill-defined, which goes beyond traditional project scheduling tools and workflow technologies.

1.1 Related Work

Several solutions have been proposed recently to deal with the coordination problem. A brief review of coordination study and systems has been presented in (Malone, 1994)(Whitfield et al., 2000)(Coates et al., 2000). A common approach to process coordination is to enforce the sequential ordering of execution steps with control conditions (Conen and Neumann, 1998). This approach mainly focuses on enactment of a process, rather than mediation between processes and actions. They define fixed processes and there are few workflow models focusing on dynamic aspects of design processes. The other work recognizes the coordination problem at ever higher levels of project management processes based on a complex network of task inter-dependencies (Malone and Crowston,, 1994) (Park and Cutkosky, 1999). These approaches model coordination as managing dependencies between activities, but lack of the capability of relating dependency information to individual activity states. My work is an extension of the work in (Jeng, 1999) and has similarities with those developed by (Dourish, 1996). Many efforts in coordination research have taken a pessimistic approach to constraining data flow or control flow, rather than taking an optimistic approach to encourage overlapping problem solving as often needed in design (Clark and Fujimoto, 1989), as shown in Figure 1.

Figure 1. overlapping processing of shared information between upstream and downstream activities

To address this limitation, I take a rule-driven approach to modeling process knowledge for coordination of distributed activities. In this paper,

coordination knowledge is represented by a set of dependency rules that determines when, how, and by whom each activity is performed. A design coordination model called DCM is introduced in terms of the dependency rules and the activity states associated with the use of the whole range of computer applications. DCM permits coordination flexibility and allows visibility of coordination logic to deal with dynamic aspects of design processes. A web-based system prototype that implements the design coordination model is demonstrated. Opportunities for using the system prototype to aid design coordination are discussed.

1.2 A Design Coordination Example

In order to demonstrate the methods of coordination presented in this paper, I apply them to a design coordination example. Here I draw a simple example from school design, involving interaction of the design architect and the energy consultant. For this example, I consider only a few aspects, with a focus on capturing the process knowledge to aid project coordination.

An architectural firm's design architect is developing the schematic geometry for different types of school buildings. In parallel, the architectural firm has employed an energy consultant, in part dealing with ecological issues. The consultant is to assess environmental conditions and energy efficiency for green building design. In order to speed up the design process, they have tacit agreement to partially overlap their work while both groups develop their own detailed schedule independently. To do this, they must closely coordinate based on a shared understanding of the process and the dependencies under changing conditions. To provide timely feedback, the energy consultant requests for the initial layout from the architect while the architect is working on the building layout. When the architect is asked for design drawings, she passes a copy of the partial building layout and then continues to detail classrooms. In the meanwhile, the energy consultant examines the building layout and analyzes potential ecological issues. During the analysis, timely feedback is passed from the consultant to the architect before assessing the behavior in lighting simulation. The revision is quickly done by the architect after receiving the feedback from the consultant. However, according to the organizational discipline, the architect cannot release formal data to the client and the engineers until she has the firm's lead architect review the drawings. The lead architect is notified of the work progress as soon as the building layout is done. After the lead architect approves the work, the drawings are passed again to the consultant, who starts to run a lighting analysis application without further design iterations.

In the examples, I review some salient characteristics of most coordination processes in current architectural practice, in a manner supporting formalization of coordination processes. The example addresses several aspects of design processes, including (1) review and approval processes; (2) tentative commitment with early release of preliminary design information; (3) management of hidden delays and bottlenecks; (4) audit trail of task decomposition and process changes. Traditionally, these processes are coordinated in a sequential order. This paper argues that the activities between upstream and downstream processes must be overlapped, as shown in Figure 1. I return to the example after the technical details of the work are presented.

2. DESIGN COORDINATION MODEL

In order to determine if the existing process modeling approaches could deal with the coordination issues, it is first necessary to identify the basic concepts, abstractions, and constructs needed in representing design coordination processes. Here, I develop a design coordination model called DCM. DCM is rule-based, with axioms defining the process dependencies between activities.

2.1 Base Elements

DCM is based on a small number of structures to capture meaningful process semantics. I define five types of primitives:

1. *Activities* are performative conditions of the ongoing execution of the subset of a task undertaken by multiple designers. An activity represents a unit of work, a unit of responsibility, and a unit of ownership.
2. *States* are tags associated with an activity. States are normally defined in terms of work progress specified by designers to effectively realize design work. The state serves as a flag for notifying other interacting activities for collaboration.
3. *Events* are significant occurrences or a set of named operations. Events may be internal, as system operations imposed by scheduling requirements or organizational policies. Alternatively, events may be raised externally, as user actions specified by designers. An event causes the activity move from one state to another.
4. *Dependencies* are semantic process relations. There are two kinds of dependencies. One defines the precedence order of activities. The other defines the synchronization of activities.

5. *Protocols* are a set of rules required for maintaining the semantic relations between activities. A protocol formalizes tacit process knowledge for defining an interaction policy.

Based on the above definitions of the basic concepts, I start to represent design activities, dependency rules, and coordination protocol, in a manner that makes the logic of coordination process apparent and tractable.

2.2 Representing Design activities

DCM represents design activities in a state transition diagram, corresponding to the execution of the subset of a task. The DCM model does not represent internal design actions or operations in detail. Rather, it deals with those aspects of an activity that are controllable and visible to others.

The state transition diagram is shown in Figure 2. The node represents the activity state and the arc represents the event. An activity specification is transformed by events through a sequence of states. A series of significant events lead to transitions between the activity states, including *start, finish, stop,* and *resume*. A set of activity states that are visible to others in executing a task, include *inactive, active, done, prepare to start, prepare to finish, not ready, pending,* and *not complete*.

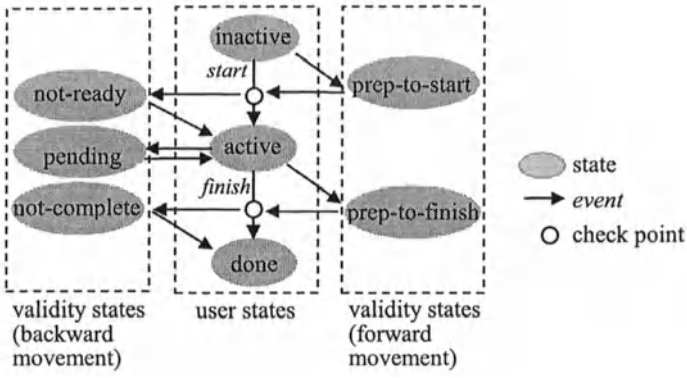


Figure 2. A process model represents the movement of the activity state along a course of events

In order to preserve user autonomy, I separate the activity states into two groups: *user states* and *validity states*. *Inactive, active,* and *done* are user states controlled by designers to reflect individual work progress. If one selects a computer application to execute the task, the user state may reflect the status of the execution of the external application. "Inactive" is the initial state before one starts up the execution process. It moves onto the "active" state when one starts to execute the task. The individual remains in

the "active" state until the task is finished. One may move to the "pending" state if any coordination is required. When one finishes the task, it moves onto the final state "done".

Another group is validity states that define the global validity of the activity. The global validity of the activity is determined based on the evaluation of the dependency constraints associated with it. Validity states include forward movement (i.e. *prepare to start, prepare to finish*) and backward movement (i.e. *not ready, pending,* and *not complete*). An activity is *not ready* to start if the dependency constraint does not hold. The activity ought to *prepare to start* when the activity on which it depends has finished. The *pending* state prevents the activity from being active or continuing. The activity is *not complete* when the dependency constraint does not hold. The activity ought to *prepare to finish* when it is required by other dependent activities.

A design activity throughout the process can be regarded as a continuous state movement from the *inactive* state to the *done* state. Every designer wants to get their work *done*. However, the path of movement cannot be optimized in collaborative design processes. During the state transition, the activity state must be checked by dependency constraints for global validity. The actual progress is then realized by combining the user states and the validity states.

2.3 Specifying Dependency Constraints

DCM specifies two kinds of dependencies: $(e1_A \rightarrow e2_B)$ and $(e1_A < e2_B)$. The former constrains the co-existence or synchronization of activities and the latter constrains the precedence order of activities. DCM uses two sets of operators to specify the dependencies:$\{\rightarrow, \leftarrow\}$ and $\{<,>\}$. The synchronization constraint can be represented in the form of $(e1_A \rightarrow e2_B)$ where $e1_A$ denotes the event *e1* acting on the activity *A* and $e2_B$ denotes the event *e2* acting on the activity *B*. The synchronization constraint $(e1_A \rightarrow e2_B)$ denotes that $e2_B$ must occur in the presence of $e1_A$. If the event $e1_A$ never occurs, the occurrence of $e2_B$ is invalid.

The second set $\{<,>\}$ constrains precedence order of activities by comparing their temporal values.The precedence constraint $(e1_A < e2_B)$ denotes that $e1_A$ must occur before $e2_B$. If $e2_B$ occurs before the event $e1_A$, the event $e2_B$ is invalid until the occurrence of the event $e1_A$.

The dependency structure can be expanded by using logical implication. One or more dependencies imply other dependencies using the transitive property of the dependencies. For example, if the dependencies $(finish_A \rightarrow finish_B)$ and $(finish_B \rightarrow finish_C)$ hold, then the dependency $(finish_A \rightarrow finish_C)$ is implied. The complexity of dependencies can be

expanded by adding the negation operator (i.e. ¬) to the dependency constraints. One possible use of the negation operator is to constrain the occurrence of an event. For example, the dependency ($start_A \rightarrow \neg start_B$) can be useful for concurrency control where B cannot start if A has started. A more complex relation can be expressed using conjunction/disjunction between two events. For example, the dependency $start_C \leftarrow (start_A \vee start_B)$ describes that C must start if A or B starts. Alternatively, we can have the dependency $start_B \leftarrow (start_A \wedge Start_C)$ describes that B must start if both A and C start. In these cases, we can consider $(start_A \vee start_B)$ and $(start_A \wedge start_C)$ to be predicates whose relations are stored in the database. For simplicity, this paper does not describe the logical implication in detail. I am working on another paper for formalization of the logical properties of the dependency relations.

3. RULE-DRIVEN COORDINATION PROTOCOL

An important aspect of the DCM model is the coordination protocol that defines the interacting behavior of the coordination processes using dependency constraints. A coordination protocol explicitly incorporates two kinds of process semantics: (1) interaction patterns regarding dependencies between activities; (2) the rules of coordination that the events and the activities must satisfy. Rather than attempt to define a fixed protocol, DCM instead specifies the protocols abstractly in terms of the dependency constraints that the behavior of the activities must hold. DCM uses such dependency rules as assertions to define an open coordination protocol, as shown in Figure 3.

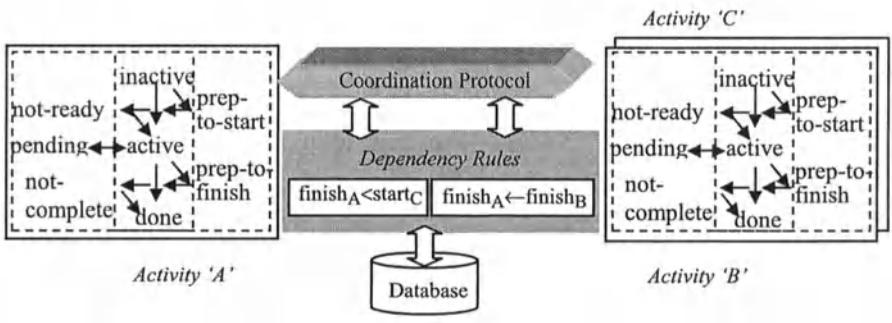


Figure 3. A rule-driven coordination protocol constrains the precedence order and the co-existence of activities

The coordination protocol suggests when people should do what during design so that at the completion of work, all team members maintain a certain working dependency. The protocol does not enforce, reject, or delay

the occurrence of the event. Consider a common example that the activity *C* needs information from its predecessor A to start his work. In order to deal with the interaction between *A* and *C*, one may define a coordination protocol through which data is transferred. The coordination protocol for the request-offer relationship is expressed in terms of two dependency rules: $finish_A \rightarrow finish_B$ and $finish_B < start_C$, as shown in Figure 3. The former describes that the activity *A* cannot finish unless the interaction activity *B* has finished. The latter says that *C* can start only after *B* has finished. This implies that, if *B* is not finished for some reasons, for example, because the data is not correctly or completely transferred, then *A* has not yet finished and *C* would not be able to start. Note that the coordination protocol is defined by the dependency constraints. The dependency constraints can be dynamically imposed to change working relationships under changing conditions. The dependency constraints are stored in the server database as process management components distributed over Internet. Details about the implementation of the system prototype are described in the next section.

4. THE SYSTEM PROTOTYPE

4.1 The Trigger Mechanism

In DCM, dependency constraints are implemented by using *database triggers*. A trigger consists of three components: *event, condition,* and *action*. This is well known as Event-Condition-Action (ECA) rules in the database field. The event specifies *when* the trigger is raised. The condition is evaluated after the trigger is raised. The condition is an expression that produces a Boolean result upon evaluation. If the condition holds, the action is then executed.

With the specification of ECA rules, I define the trigger for evaluation of the dependency constrains in DCM model. I refer to it as dependency rules. A dependency rule has the form as follows:

WHEN	*an event occurs*
IF	*the dependency constraints satisfy*
THEN	*set the activity state*

where the "*WHEN*" clause specifies the user event (e.g. *start, finish*) that causes the raising of the trigger; the "*IF*" clause specifies the condition of dependency constraints (e.g. $finish_A < finish_B$); the "*THEN*" clause sets the activity is in a particular state (e.g. *inactive, not-ready, prep-to-start, active, pending, prep-to-finish, not-complete, and done*). The "*IF*"condition is used

to control whether or not the trigger action is performed. In many cases, the "*IF*"condition can be optional specification. If it is omitted, the trigger is called unconditionally.

Triggers are generally used in the database field to perform parallel database updates and enforce integrity constraints. Here I borrow the notion of ECA rules to specify dependency rules. The dependency rules are evaluated during the processing of a user event. The triggers are used as a mechanism to manage the movement of activity states without changing the behavior of user activities.

4.2 Applications to The Design Coordination example

In order to demonstrate the methods presented here, I apply them to the example developed in Section 1.2. I begin by assuming that the overlapping process is decomposed into smaller process items, whose scope allows more semantically specification of dependency rules. The detailed precedence chart of the processes is presented in Figure 4.

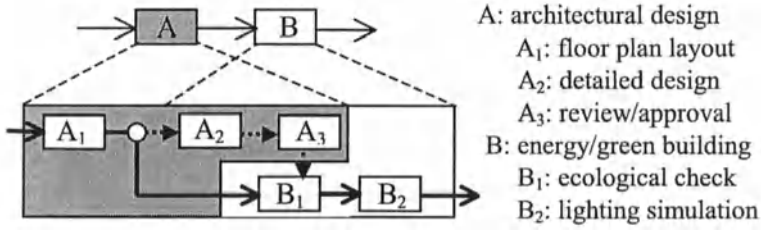


*Figure 4.*The process is decomposed into smaller process items, whose scope allows more semantically specification of dependency rules.

Some definitions of the dependencies used in the example are shown in Table 1. Assume that the design architect is working on the floor plan layout and the activity state of A_1 is initially set to *active*. As the work of the floor plan layout is finished, the design architect uploads the drawings and submits a "*finish*" event to the server database. The procedure is done by filling out a form on a web page. When the form is submitted, DCM creates a process instance. Given the process instance (i.e. "$finish_{A1}$"), the rule *R1* is triggered to reset the activity state of B_1 to "*prep-to-start*". If the energy consultant B_1 logs on to the web page, he will be requested to prepare to start his work. At the same time, the process instance $finish_{A1}$ also raised the rule *R2* that checks if the condition (i.e. $start_{A1}<finish_{A3}$) holds. Since the lead architect has not reviewed the drawing yet, the system resets the activity state of A_1 to "*not-complete*", preventing A_1 from completion. A_1's work will not be complete until the lead architect finishes his review process on the drawings. The activity state of A_1 may be reset to *done* immediately after the

design architect finishes his review process, which is done remotely in our system prototype through Volo View over Internet.

Table 1. Listing of the dependency rules used in the example

R1	WHEN	($finish_{A1}$)
	IF	(B_1.state = "inactive")
	THEN	(B_1.state = "prep-to-start")
R2	WHEN	($finish_{A1}$)
	IF	($finish_{A1}$<$finish_{A3}$)
	THEN	(A_1.state = "not-complete")
R3	WHEN	($start_{B1}$)
	IF	(A_3.state = "active")
	THEN	(A_3.state = "prep-to-finish")
R4	WHEN	($start_{B2}$)
	IF	($start_{B2}$<$finish_{B1}$)
	THEN	(B_2.state = "not-ready")

To encourage overlapping problem solving, the early release of design information by A_1 is accessible to the energy consultant in the interim between the *"not-complete"* state and the *"done"* state. As a design collaborator, the energy consultant may access the drawings so as to have instant feedback and start ecological checking earlier. Yet, the energy consultant assesses the situation by tracking the dependencies and realizes that the final release is dependent on the completion of the review process performed by the lead architect. With shared understanding of the dependencies between distributed activities, the energy consultant will take the responsibility for any design actions taken on early release of design information.

In parallel, when the energy consultant starts up the work on ecological issues, the rule R3 is triggered. The activity state of A_3 is set to *"prep-to-finish"*. This will notify the lead architect that he should prepare to finish his work. The same principle is applied to other downstream activities. If B_3 starts and B_1 has not finished, the activity state of B_3 will set to *"not-ready"*, preventing B_3 from starting the work. The dependency rules can be used optimistically to prevent hidden delay, or pessimistically to prevent from wasting time to work on incomplete input data or the data that is possibly to change.

4.3 The Implementation

In the prototype implementation, I chose MS SQL Server as the backend server database, and use Java and ASP to implement the web interface.

Dependency rules are specified on a web page and stored in the form of triggers in the server database. In the example, I use AutoDesk Volo View as the frontend application to implement whiteboard markup capabilities. The markup scheme is semi-automatically passed over and stored in the server. Mappings between the application and the databases are coded in various dynamic-linked libraries using Java and XML. The system prototype is shown in Figure 5, including an overview of activity states, a process structure, a graphical viewer, and a message board for discussion. A warning message shows that a dependency constraint is violated when one wants to start up ecological checking.

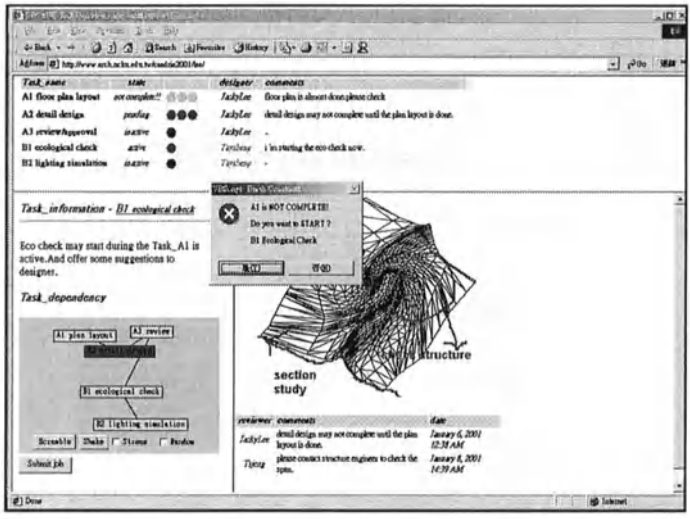


Figure 5. A warning message pops up when a dependency constraint is violated.

5. CONCLUSION

Collaborative design has imposed an essential requirement on coordinating multiple interacting processes to support a new way of working, especially with the increasing use of the whole range of computer applications. In this paper, I have developed a modeling tool to facilitate design coordination. The DCM model has several advantageous capabilities that are not supported in current collaborative design environments. First, DCM is *application-independent*. DCM model overlays activity states on a process structure, which serves as a logical basis for coordination of distributed activities. The state transition model captures the process semantics of an activity, which is independent from any applications. Secondly, DCM provides *an open coordination protocol.* Many of the

dependencies in designing are not deterministic, but can be resolved in a variety of ways. Our work separates the dependency constraints from the sequential ordering of design activities. Dependency constraints can be dynamically asserted to the system, allowing the open-ended solutions to the design coordination problem. Thirdly, DCM permits *coordination flexibility*, allowing defining and articulating varied processes and their dependencies in a partial order. Rather than enforcing the dependencies in a strict manner, the coordination protocol checks the validity of activity states and allows a certain degree of tolerance for violation of dependency rules. At the completion of work, all team members must maintain a certain working dependency.

This paper attempts to facilitate coordination of distributed activities using computer augmentation. An important goal of this research is to off-load some of the coordination responsibilities from designers in order to reduce the apparent complexity and the effort involved in collaboration. Future work is to use the same principle to a large coordination process for multi-organization collaboration.

6. REFERENCE

Conen, W. and G. Neumann [eds.], 1998, *Coordination Technology for Collaborative Applications*, Springer.

Clark, K. B. and T. Fujimoto, 1989, "Overlapping Problem Solving in Product Development", *Managing International Manufacturing*, in: K. Ferdows (ed.), Elsevier Science Publishers.

Coates, G., R. I. Whitfield, A. H. B. Duffy, and B. Hills, 2000, "Coordination Approaches and Systems- Part II: An Operational Perspective"", *Research in Engineering Design*, 12, p. 73-89.

Dourish, P., J. Holmes, A. MacLean, P. Marqvardsen, and A. Zbyslaw, 1996, "Freeflow: Mediating Between Representation and Action in Workflow Systems", in: *Proceedings of the ACM Conference on Computer Supported Cooperative Work CSCW'96*, Boston, USA.

Eastman, C, I. His, and C. Potts, "Coordination in Multi-Organization Creative Design Projects", GVU report, College of Computing, Georgia Institute of Technology, 1999.

Jeng, T., 1999, *Design Coordination Modeling: A Distributed Computer Environment for Managing Design Activities*, PhD Thesis, Georgia Institute of Technology, January, 1999.

Malone, T. W. and K. Crowston, 1994, "The Interdisciplinary Study of Coordination", *ACM Computing Survey*, 26(1), March.

Park, H. and M. R. Cutkosky, 1999, "Framework for Modeling Dependencies in Collaborative Engineering Processes", *Research in Engineering Design*, Vol. 11, p.84-102.

Whitfield, R. I., G. Coates, A. H. B. Duffy, and B. Hills, 2000, "Coordination Approaches and Systems- Part I: A Strategic Perspective"", *Research in Engineering Design*, 12, p. 48-60.

Capturing Histories of Design Processes for Collaborative Building Design Development

Field Trial of the ADS Prototype

Cristina Cerulli, Chenghzi Peng and Bryan Lawson
University of Sheffield

Key words: Design Rationale, Design Support Systems, Usability Evaluation

Abstract: The ADS Project - Advanced Design Support for the Construction Design Process - builds on the technological results of the *previous* COMMIT Project to exploit and demonstrate the benefits of a CAD based Design Decision Support System. COMMIT provides a system for storing knowledge about knowledge within the design process. It records design decisions, the actors who take them and the roles they play when doing so. ADS links COMMIT to an existing object-oriented CAD system, MicroStation/J from Bentley Systems. The project focuses on tackling the problem of managing design information without intruding too much on the design process itself. It provides the possibility to effectively link design decisions back to requirements, to gather rationale information for later stages of the building lifecycle, and to gather knowledge of rationale for later projects. The system enables members of the project team, including clients and constructors, to browse and search the recorded project history of decision making both during and after design development. ADS aims to facilitate change towards a more collaborative process in construction design, to improve the effectiveness of decision-making throughout the construction project and to provide clients with the facility to relate design outcomes to design briefs across the whole building life cycle. In this paper we will describe the field trials of the ADS prototype carried out over a three-month period at the Building Design Partnership (BDP) Manchester office. The objective of these trials is to assess the extent, to which the approach underlying ADS enhances the design process, and to gather and document the views and experiences of practitioners. The ADS prototype was previously tested with historical data of real project (Peng, Cerulli et al. 2000). To gather more valuable knowledge about how a Decision Support System like ADS can be used in practice, the testing and evaluation will be extended to a real project, while it is still ongoing. The *live case study* will look at some phases of the design of a mixed

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 427–437.
© 2001 *Kluwer Academic Publishers.*

residential and retail development in Leeds, UK, recording project information while it is created. The users' feedback on the system usability will inform the continuous redevelopment process that will run in parallel to the *live case* study. The ADS and COMMIT Projects were both funded by EPSRC.

1. INTRODUCTION

1.1.1 The ADS project

The ADS Project (Advanced Design Support for the Construction Design Process), funded under the Innovative Manufacturing Initiative by the EPSRC, aims to exploit and demonstrate the benefits of a CAD-based Design Decision Support System. The project builds upon earlier work on the more theoretical information management concepts developed in the COMMIT (Construction Modelling and Methodologies for Intelligent Information Integration) project, an earlier EPSRC funded project.

ADS develops naturally from COMMIT by incorporating the advanced information management and decision support techniques of the COMMIT project into an existing object-oriented CAD system (Microstation/J from Bentley Systems), and applying this tool to the management of design information and decision making in a real life project provided by Building Design Partnership (BDP), a large construction design practice.

In implementing the COMMIT approach in a real design situation, the emphasis shifts towards learning and understanding more about the decision-making process within design activities. The motivation behind the COMMIT and ADS projects is concerned with defining mechanisms to handle the proactive management of information to support decision-making in collaborative projects.

ADS focuses on how to provide designers with tools for recording and managing the group dynamics of design decision-making in a project's lifetime, with the explicit intention to minimise any intrusion on the design process itself.

The deliverable of the ADS Project is an advanced CAD tool that facilitates capturing designers' rationales underlying their decision-making throughout the design and construction development. The system also enables members of the project team (extendable to all the actors involved in the process, including clients), to search and browse the recorded project history of decision-making, during and after design development.

1.1.2 Commit and beyond

As mentioned above, the ADS project was set up to bring forward the developments from the COMMIT project which was concerned with the management of information to support decision making in multi-actor environments. It addressed six primary issues that are central to information management:

1. The handling of ownership, rights and responsibilities;
2. Versioning of information;
3. Schema evolution;
4. Recording of intent behind decisions leading to information;
5. Tracking of dependencies between pieces of information;
6. Notification and propagation of changes.

Many of these are distinct issues, but they have been found to be closely inter-related, making it difficult to address them individually. During the COMMIT project, the Salford group has employed object-oriented technologies (first in C++ then in JAVA) to implement an information management framework that addressed the problems indicated above (Brown, Rezgui et al. 1996).

COMMIT does not impose a decision making sequence, leaving it to the design team, but provides an infrastructure through which all members of the team have the opportunity to be aware of what decisions were made, who made them and when as well as why. The way in which this is achieved is described elsewhere (Rezgui, Cooper et al. 1998).

1.1.3 ADS: aims and objectives

The key aim of the ADS project is to develop a system that is adequate to demonstrate an object-oriented approach to managing design decision-making across the whole building life cycle.

The ADS research project also offers an opportunity to investigate a number of issues concerning computer-mediated collaborative design processes such as the integration of recording/capturing design intents/rationales into a general CAD platform.

With ADS, designers are provided with the tools to record any information related to a particular design decision-making process. That information can then be recalled and accessed by other actors involved in the process such as clients, other designers, contractors etc.

At any point in time the actors involved in the process are enabled to make informed design decisions in the light of the information about other design decisions related to the *current* one, that are being or have been made

by other project actors. The system supports and facilitates the collaborative asynchronous decision-making process.

1.2 ADS field trials

Following a pilot Case Study in which ADS was populated with historical data (Peng, Cerulli et al. 2000), ADS is currently being tested on live projects with the collaboration of the Manchester office of BDP. The projects used for the trials are: the *Round Foundry Residential and Retail Development* in Leeds and the *Deansgate Hotel* in Central Manchester. ADS is being used to record design decisions as they are made over a 3-month segment of the design process. Currently two of the designers involved in these projects are using ADS.

During the field trial every attention was paid to avoid any interference with the design development as well as any imposition regarding the frequency at which to insert data in the system. Designers regularly e-mailed the updated ADS project database and the model dgn files; short meetings were periodically held to gather feedback about system and interface usability and for post hoc interviews about the data analysis.

One of the main objectives of the ADS field trials is data gathering. Populating the system with real data gathered in real time, in anger, without any artificial simplification of the design development process.

Associated with this objective is the intention to explore the potential of ADS as a tool for carrying out research on design processes as an unobtrusive way to monitor real design processes, without significantly interfering with the observed process. Lawson identifies five methods of investigating design processes: speculating about design, laboratory observation of designers under rigorous empirical conditions, observing designers at work in the studio, listening to designers telling about the work they do, either by interviewing them or reading what they have written about their process and simulating the design process (Lawson 1997). All these ways of researching design processes have been tried and each appears to have some flaws. Either the events studied do not reflect real events or the analysis is bound to be biased by the investigator's personal perceptions or the experiments deal with artificially limited phases of design or the fact that knowledge about the process often remains implicit in the designer's head. Despite the ADS system was originally developed as an innovative tool for supporting decision making in design (Cooper, Rezgui et al. 2000; Peng, Cerulli et al. 2000) the research group realized that it could also offer a fundamentally new methodology for studying the design processes by capturing design development events in a relatively unobtrusive way.

Other key objective of the ADS field trials is the evaluation of both the ADS System and the User Interface. This objective partially clashes with the previously stated aim of non-obtrusiveness in data gathering, but is seen as a necessary stepping-stone towards a more usable system. It has to be pointed out that these field trials are regarded as a tool to support the system development: user feedback and evaluation, as well as results of the project data analysis will feed back directly into the development that runs in parallel to the experiment. Incremental changes to the system will be continuously implemented and released for testing and evaluation. A few development cycles will be iterated throughout the duration of the case study.

The project data gathered by the designers through ADS is being analysed as it is produced, and post-hoc interviews are held with the participants to confirm any interpretations of the data that arise from the analysis. For an account of the completed ADS Live Case Study see (Cooper, Rezgui et al. 2001).

1.3 ADS Field Trials - Phase I

1.3.1 Description of the ADS prototype as currently released

Designers were given the ADS tool to work on the Round Foundry and Deansgate Hotel Projects and they were asked to generate ADS Design Decision Records as a result of design decisions.

An ADS Design Decision Record is a complex dataset formed by many types of data: CIMM management attributes; ADS taxonomical attributes; CAD transaction; description of design intents/rationale; affected and pending design decisions; and hyperlinks to other related documents. For a detailed account of the ADS Decision Record please refer to (Peng, Cerulli et al. 2000).

In this phase of the live case study the system was capable of recording only the following data types:

- The CIMM management attributes. The CIMM Manager deals with the information regarding "people" involved in a building design project. Members of the project can generate a member profile by addressing the following attribute: Actor (Title, Name, Phone, Email, Fax, etc.); Role (The roles as played by the members of the project team); Default Authorities (Object Type, Role, Methods); Specific Authorities (Object Instance, Role, Methods).

- CAD transactions. All the identities of the CAD elements or components involved in any transaction as recorded by the ADS Event Listener. The transaction can span over many work-sessions and across multiple files.
- Description of design intents/rationale. Input of textual descriptions in a natural language (English) of what the designer thinks are the reasons behind a design decision or design changes. Intent representation promotes the Actor's understanding of the reasons for project changes, which in turn reduces problems arising from misunderstanding intentions.

The decision record generation process can be summarised as follows: the designer logs on her client workstation and gets access to a project CAD design file stored in the project's server workstation. When a number of changes have been made as the result of a design decision, they may be committed as a single transaction, with a brief explanation of the rationale for that decision. This has some resemblance to filling in a change box in a paper drawing, but it can encompass a change that would be represented on more than one drawing, and, most notably, there is total freedom of choice regarding the granularity at which this information is recorded.

Before committing a decision the system checks to see which previous decisions may have been affected by the current changes (by inspecting the changed elements). The *Commit* Decision Browser displays as highlighted the past decisions that have been affected by the changes in the current transaction (*Figure 1*) and it is possible to browse information relative to those previous decisions such as actor, time, rationale, CAD objects involved etc. For objects that are able to be represented graphically in the CAD tool, this facility also allows the user, by means of the *Select Objects* button, to view the different versions of an object graphically in context within a window of the CAD tool.

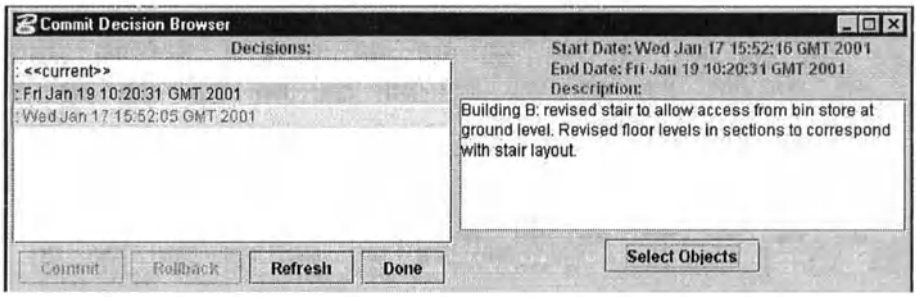


Figure 1. The Commit Decision Browser

1.3.2 Data gathered

A number of design decisions were recorded into the system. To illustrate the type of data gathered let us consider one of the decisions recorded during Phase I (*Figure 2*) committed by the actor *Garrett, S.* in her role of *Architect,* to which the *rights* of creating/deleting/modifying the model had been assigned.

The user was left totally free to determine at what point to commit a decision, the amount of information to insert and the number of design changes to be included in a single decision or transaction. The rationale for that decision was input in an unstructured form in a free-text box, and, for the decision in examination, reads as follows: "Building B: revised stair to allow access from bin store at ground level. Revised floor levels in sections to correspond with stair layout". The system also stored information about the CAD elements (dgn objects) involved in that decision. They belonged to two different dgn files: ap0120_02.dgn; a plan, and ascc20_02.dgn; a section (the relevant portion of those files is shown in *Figure 2*). The *Select Objects* button allows highlighting the CAD object involved in the decisions when one of the files containing them is open (MicroStation does not allow having more than one dgn file open at the time).

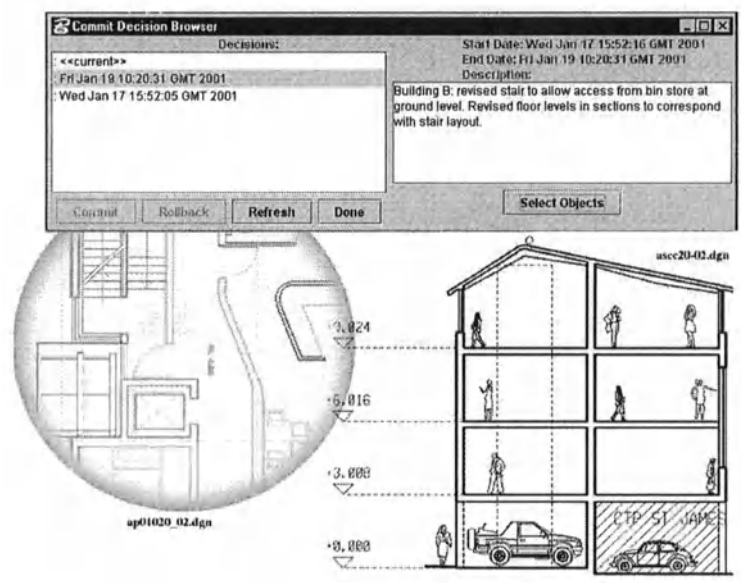


Figure 2. Illustration of a design decision

1.4 ASD field trials - phase II

Throughout the field trials the software development proceeded in parallel with the use of the system. Features that had been previously designed have been implemented and the feedback on system and interface usability gathered from interviews and e-mail communication between designers and researchers was taken into account. A new version of ADS is about to be released and it will mark the beginning of Field Trials - Phase II.

1.4.1 Description of the ADS phase II prototype

The main features of the Phase II ADS release are an improved User Interface, based on the results of evaluation, and the implementation of extra data gathering functionalities. In particular the additional data types that can be gathered and displayed through the *Commit Decision Browser* are hyperlinks to related information and rationale taxonomical attributes.

1.4.1.1 Hyperlinks to related documents

A design decision may be made in relation to other information that is stated in other documents such as minutes of a project meeting with the client, a WWW page showing a building component product, etc.

Gruber and Russell observed that rationale explanations are often inferred from information that is stored, rather than being stored as exhaustive explanations (Gruber and Russell 1996, p. 330). The possibility to hyperlink the decision record to related documents will therefore enrich the contextual information about that decision, reducing the amount of information that need to be input s for that decision.

1.4.1.2 Taxonomical attributes.

The idea of establishing a *taxonomy of design decisions* emerged from our research into the possible ways of capturing design rationales from designers' actions performed on a CAD platform and into what type of design information is more likely to be useful once it's been captured. Gruber and Russell, from a protocols analysis of designers talking about design, derived a taxonomy of categories of information requested about designs (Gruber and Russell 1996, p. 326). It seemed to us that we might experiment with a generic classification scheme that may cover most, if not all, kinds of design decision-making in the processes of building design. As a research hypothesis to be further verified, we consider that a taxonomy of design decision is useful because for:

a) Post Analysis. Design intents or rationales entered by the designers through the taxonomical interface can generate data records that will allow for further analysis after a project is completed.
b) Data Gathering Efficiency. A well-defined and designed taxonomy can enrich the user interface so as to help the user to think about design rationales at the time of making those decisions. A taxonomy can provide efficiency in recording design rationales if the designer does not have something specific to start with at the time of making the decisions.
c) Data Structure: Each attribute of the taxonomical structure (template) can be used as a data entry to index and characterise design decisions so to improve the organisation and search of recorded decisions.

The provision of a taxonomy of design decisions within the ADS framework is therefore mainly to facilitate the generation and use of design decisions during and after design development. It has to be pointed out, though, that the taxonomy provided could only be an open-ended scheme in which the end users can define and extend the scope at any one time to reflect the demands of the project as it develops. We also understand that design decisions may not be attributed in simple or general taxonomical terms, and the system should aim to preserve the "context" as much as it can.

The decision to build the ADS taxonomical attributes in the data gathering interface was encouraged by the findings from the analysis of the free text description of the rationales for decisions committed in Phase I and by interviews with the designers.

2. CONCLUSIONS

The ADS system as a recording tool is still under testing and under development. The data structure proved to be versatile, easily accommodating changes and developments in the software architecture.

Minimum intrusiveness is crucial to the success of any decision support and design rationale-gathering tool. With ADS the granularity of decisions is determined entirely by the designer using the system. Obviously the benefits deriving from recording design rationale are proportional to the quantity of data gathered and, possibly, inversely proportional to the granularity of events/decisions/transactions. A potential impediment or deterrent to the data gathering is the fact that it is likely that the main beneficiary of such activity is not going to be the very person that is requested to input the data into the system. But there is also a cultural dimension of the construction design process that determines the success of design rationale and project information capturing. It is possible to envision a gradual increase in the

amount of information recorded into the system as the designers become more aware of the real potential benefits of recording design rationale.

The user response was very sympathetic towards the overall objectives of the system. Frustration was occasionally expressed towards the limitations of specific implementations. In particular limitations in processing speed were pointed out as disruptive and intrusive. These issues are currently being addressed to be included in the next release of ADS.

In this phase of the field trials, fine-tuning to the user needs the system data recording functionalities necessarily shifted the focus on the ADS system as a data gathering tool rather than a design aid tool. In the next phase of the field trials and development, still ongoing, more emphasis will be placed on improving the retrieval of information. Extra functionalities will be implemented like the notification of changes to potentially affected objects, mechanism for mapping relationships between decision (affected and pending decisions) and the nesting of design decisions.

An account of the completed ADS Live Case Study and a critical evaluation of the system will be published in (Cooper, Rezgui et al. 2001)

3. ACKNOWLEDGMENTS

The Advanced Decision Support for the Construction Design Process project is funded by the UK Engineering and Physical Sciences Research Council (EPSRC), Innovative Manufacturing Initiative (Ref#GR/M42398). Building Design Partnership participated in the Field Trials and provided documentation about the Round Foundry Project in Leeds and the Deansgate Hotel in Central Manchester. Bentley Systems provided Microstation/J CAD software and support for integrating the ADS system into the CAD tool.

4. REFERENCES

Ball, L. J., N. J. Lambell, et al. (1999). "Representing design rationale to support innovative design re-use: a minimalist approach". *4th Design Thinking Research Symposium on Design Representation*, Cambridge, MA, MIT.

Brown, A., Y. Rezgui, et al. (1996). "Promoting Computer Integrated Construction Through the Use of Distribution Technology." ITcon 1: 1-16.

Cooper, G., Y. Rezgui, et al. (2001). "CAD-Based Trials of an API for Decision Support". *Construction Information Technology CIB W78,* Mpumalanga, South Africa.

Cooper, G., Y. Rezgui, et al. (2000). "A CAD-Based Decision Support System for the Design Stage of a Construction Project". *5th International Conference on Design and Decision Support Systems in Architecture and Urban Planning*, Nijkerk.

Gruber, T. R. and D. M. Russell (1996). "Generative Design Rationale: Beyond the Record and Replay Paradigm". *Design Rationale: Concepts, techniques and use.* T. P. Moran and J. M. Carroll. Hillsdale, NJ, Lawrence Erlbaum Associates: 323-349.

Lawson, B. (1997). *How Designers Think. The Design Process Demystified.* Oxford, Architectural Press, Butterworth-Heinemann.

Moran, T. P. and J. M. Carroll, Eds. (1996). *Design Rationale: Concepts, techniques and use.* Hillsdale, NJ, Lawrence Erlbaum Associates.

Ormerod, T. C., M. John, et al. (1999). "Desperado: Three-in-one indexing for innovative design". *Seventh IFIP Conference on Human-Computer Interaction - INTERACT '99*, London.

Peng, C., C. Cerulli, et al. (2000). "Recording and Managing Design Decision-Making Processes through an Object-Oriented Framework". *5th International Conference on Design and Decision Support Systems in Architecture and Urban Planning*, Nijkerk.

Rezgui, Y., G. Cooper, et al. (1998). "Information Management in a Collaborative Multiactor Environment: The COMMIT approach." *ASCE Journal of Computing in Civil Engineering* 12(3): 136-144.

An Architectural Approach to Virtual Reality Support of Multi-user Environments

Seung-Hoon Han and James A. Turner
The University of Michigan, Ann Arbor, MI, USA

Key words: Collaborative Design, Distributed Virtual Environments, Multi-user, Internet

Abstract: The Internet and its multimedia component, the World Wide Web (WWW), are the essential technological foundations, and the tools to construct cyberspace on these foundations are beginning to be created. Two of those tools are the network programming language, Java, and the 3D graphics standard for the Internet, the Virtual Reality Modelling Language (VRML) which has the ability to support programmable behavior. This paper documents at experiment with the use of networks with Java-VRML connectivity, applying it to a collaborative system which will make it possible for multiple users to navigate and dynamically update an architectural VR environment.

1. INTRODUCTION

Computer-Aided Design (CAD) has an increasingly important role in the design professions as a tool for visualizing and documenting a design solution. The increased complexity in technical support systems and the fragmentation resulting from the addition of many more individuals to a building project, however, have made the design process far too complicated. In addition, the lack of collaboration in knowledge or information representation strategies has made the communication during the design and construction phases more difficult.

CAD researches from the A. Alfred Taubman College of Architecture and Urban Planning at the University of Michigan have examined a collaborative design system, ARCH:MUVR, to support more efficient

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 439–452.
© 2001 *Kluwer Academic Publishers.*

communication and interaction in the design process. This project is based on Distributed Virtual Environments (DVE) representing multi-user virtual workspace, which have the potential to evolve the Web from a pure information space to a social space for collaborative activities (K. Saar).

Currently, there exist several multi-user environments on the Web that serve as attractive communication spaces for their clients; however, interactions within those environments are still limited and the avatars' behavior is relatively poor as compared to their visual appearance.

The research presented in this paper is focused on supporting the tasks of evaluating new distributed virtual worlds and reviewing existing shared environments. The implementation and authoring of multi-user environments has been tested through knowledge of VRML, Java, Java Database Connectivity (JDBC) and the External Authoring Interface (EAI). The development and testing of the evaluation system is an ongoing process.

2. SYSTEM OVERVIEW

The multi-user virtual space allows multiple clients, as avatars with a physical presence, to interact on the Internet. Transformations as motion updates and data modifications are transmitted to every participating client immediately.

2.1 Clients and Avatars

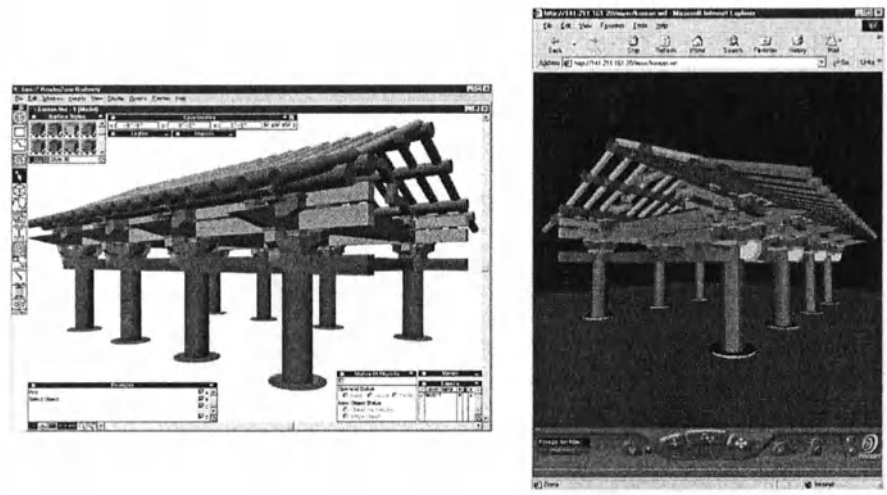
Clients are able to participate in the virtual environment and interact with other clients through avatars. A type of avatar is employed as a VRML model for the purposes of simplifying and associating multi-user interactions and is designed to represent each user. The default avatars are composed of simple models, such as spheres, cylinders, cones and boxes; however, the system could incorporate more detailed avatars allowing the client to customize his/her individual presence. As soon as the VRML file launches, the browser executes a Java class file that informs the server of the client's position in the scene and manages the other avatars in the scene.

2.2 Documents and Objects

For the virtual environment, we have tried to build interactive architectural 3D scenes - scenes that people can navigate through, can interact with, and can affect so that each visit leaves its own mark - as we do

in real world. The choice of 3D scene description language has been influenced by the fact that our primary network target is the Internet and the WWW hypermedia system it supports. A significant effort has been underway to produce an equivalent 3D description language that works well in the WWW. This language is referred to as the VRML, which provides the essential 3D building blocks to construct virtual worlds, and Java is the glue that animates the worlds and links them to the WWW.

In this project, we constructed the architectural model in form·Z and used an export option to produce the VRML model. As the theme of the multi-user space, one structure from Korean traditional architecture has been selected.



(a) Modelling in form·Z (b) VRML Model Exported by form·Z

Figure 1. Created Models for ARCH:MUVR Environment

Besides clients, avatars and virtual environments, the system also contains additional objects, such as an information panel holding connection logs, a chatting window as the communication interface, and an evaluation interface for collaborative review with Web references implemented by ColdFusion scripts, which are essentially a collection of static Web pages similar to Common Gateway Interface (CGI).

2.3 Navigation Tools

Clients navigating around the virtual world may use the standard VRML browser navigational controls, such as CosmoPlayer, CommunityPlace and

WorldView, to move around the environment. In this project, we have employed CosmoPlayer (Version 2.11) and its Java packages for EAI.

3. ARCHITECTURE OF THE ARCH:MUVR

We have employed the following five technologies in this project: 1) VRML modelling and scripting, 2) VRML communication to JAVA applets through EAI, 3) basic Database constructing, 4) Database communication to JAVA applets using JDBC, and 5) ColdFusion Markup Language (CFML) programming to support entry and processing of forms of evaluation systems.

The majority of small-scale DVEs are realized as client-server architectures since this is conceptually and practically the simplest and provides possibilities for accounting and security management (K. Saar).

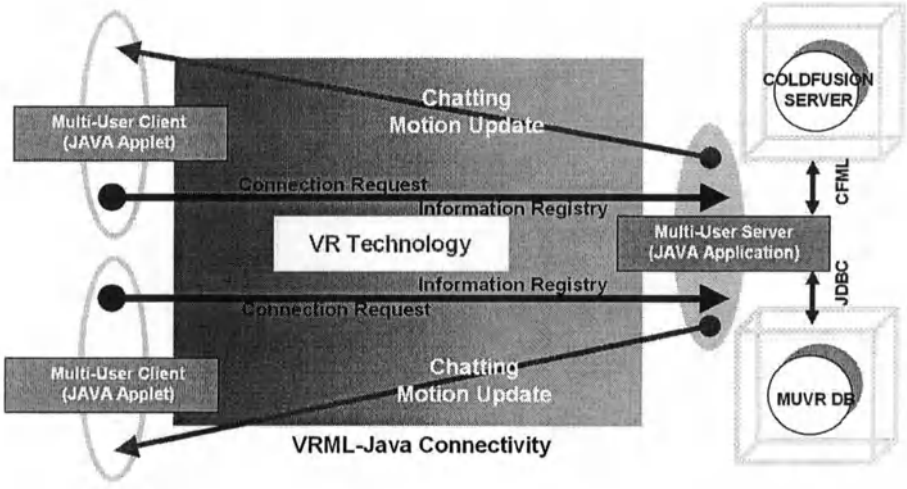


Figure 2. Technology Flow of the ARCH:MUVR

3.1 The Server Side

The structure of the network server is written in Java. When the server receives a connection request, it sets up a communication link to the client and waits for clients' requests. For each request to update the scene graph information, the server produces results and returns them back to all clients. Also, multiple threads are used to manage the connections from the clients. The use of multiple threads in Java enables the program to control

connected clients for each navigational operation. This mechanism is used for building a network server that accepts incoming connections from several clients simultaneously.

To be successfully connected to a server machine, first, a Java source file is needed to initialize connection sockets and ports in the server machine. Then, a VRML model representing a specific collaborative environment, in which a randomly moved avatar is participating, is loaded.

Once a client attempts to contact the server, it is required to enter a login name and password to perform valid operations on the system. To validate clients' logins, their information is retrieved from a database using JDBC, and one of registered avatars is assigned to a valid client.

Clients : Table

ClientID	LastName	FirstName	Login	Password	Company	Location	Position	Status	ClientIP	Avatar
1	Han	Seung-Hoon	hshoon	muvrpass	University of Michigan	Ann Arbor, MI	Architect	Connected	HSHOON/141.211.161.20	1
2	Turner	James	turner	muvrpass	University of Michigan	Ann Arbor, MI	Professor	Connected	HAN/141.211.162.95	2
3	Jones	Mike	mike	muvrpass	SOM	Chicago, IL	Designer	Disconnected	None	0
4	Terrel	Ingmann	terrel	muvrpass	Parametric Co.	Boston, MA	Programmer	Disconnected	None	0
5	Shin	Namsoo	nshin	muvrpass	Chonnam University	Seoul, Korea	Researcher	Disconnected	None	0
(Number)										0

Record: 1 of 5

Figure 3. Interactive Manipulation of the ARCH:MUVR Database

The server-side Java code is responsible for the initialization of an ARCH:MUVR system for the management of clients, and for loading avatars. It keeps all the required status information for every connected client. The server executes two threads: TransformDispatcher and ChatDispatcher as soon as a client asks for participation.

Server Side Java application

```
import java.awt.*;
import java.awt.event.*;
import java.net.*;
import java.sql.*;

class Server
{
   static LinkedList clients = new LinkedList();
   // add other class variables

   public static void main(String args[]) throws IOException
   {
      ServerSocket serverSocket, chatSocket;
      Socket clientSocket, clientChat;
      DataInputStream in; // input stream from clients
      OutputStream out, chatOut; // output stream to clients
      BufferedReader loginIn;
      FileWriter logFile; // to save logs
      // other instance variables

      // opens sockets for communication
```

```
        // initializes MUVR database
        // accepts clients' requests
        // validates clients
        // updates log files
        // updates MUVR database
        // creates a server side thread
} // Server
```

3.1.1 Thread TransformDispatcher

This code is responsible for receiving position from the client avatars and sending them to all the other connected clients. It provides the client with information about its world and takes events from the client and dispatches them to other threads. The life of the TransformDispatcher thread is finished with the termination of the server-client connections.

Server Side Java Thread Regarding Transformation

```
import java.net.*;
import java.io.*;
import java.util.*;

class TransformDispatcher extends Thread
{
   Socket socket = null;
   DataInputStream in = null;
   DataOutputStream out = null;
   LinkedList clients;
   // other instance variables

   public TransformDispatcher(Socket, LinkedList, int) // constructor
   public void run() // takes a motion and dispatches to all clients
   public synchronized void output(int, float, float, float)
   // dispatches positions
   public synchronized void output(int, float, float, float, float)
   // dispatches rotations
} // TransformDispatcher
```

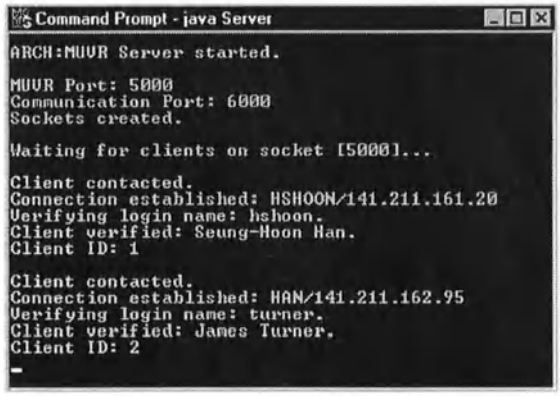


Figure 4. ARCH:MUVR Server Operation

3.1.2 Thread ChatDispatcher

This code is responsible for receiving textual information from the client avatars and sending them to all the other connected avatars.

Server Side Java Thread Regarding Textual Interaction

```
import java.net.*;
import java.io.*;
import java.util.*;

class ChatDispatcher extends Thread
{
   Socket chat = null;
   LinkedList clients;
   PrintWriter chatOut = null;
   FileWriter chatLog = null;
   BufferedReader chatIn = null;
   // other instance variables

   public TransformDispatcher(Socket, LinkedList, int) // constructor
   public void run() // takes messages and dispatches to all clients
   public synchronized void output(String) // dispatches messages
} // ChatDispatcher
```

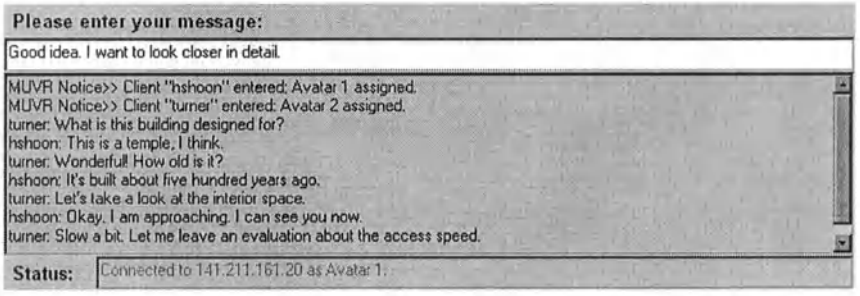


Figure 5. Textual Interaction between Clients

3.2 The Client Side

To validate clients' logins, their information is retrieved from the database and all attempts to login are recorded. In case the number of users that can be connected at the same time exceeds the limitation due to the system performance, the server will also store a message.

The client-side is designed as a Java applet and acts as the interface between the client and the virtual environment in the browser. When initializing, it establishes a connection to the server, sends login name and password, receives permission, and requests a scene description from the server. The request for an avatar and the notification of the participating clients' information, such as the name of avatar, the current position and rotation, completes the initialization process.

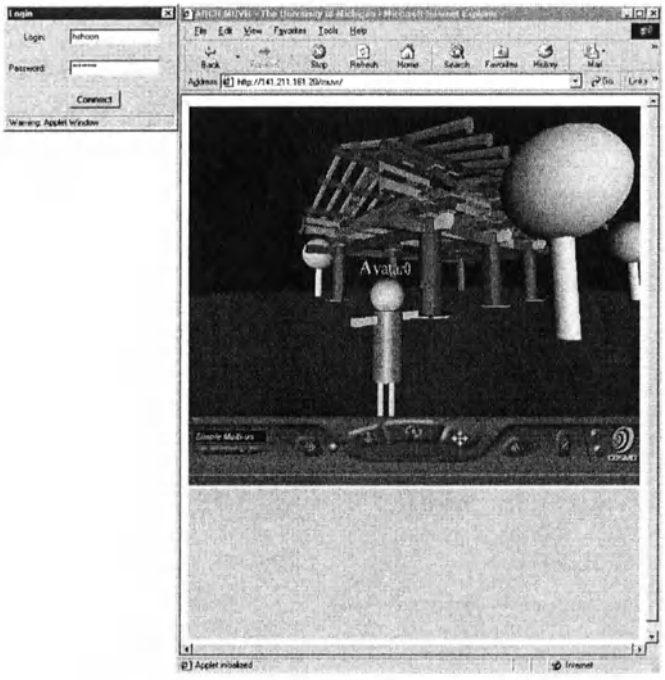


Figure 6. Client Login to the ARCH:MUVR Environment

The client-side code has the capability to manipulate the scene graph via standard EAI methods. It manages client coordinates during navigation and informs the server-side about position and rotation changes. Also, It is responsible for managing events embedded in the scene graph. For these actions, the EventOutObserver interface implemented by the callback method watches for all VRML events.

When a client terminates the connection with the server, the client-side deletes the avatar and returns all ownership to the server, and the server-side informs other threads about the disconnection of the client and updates the fields regarding clients' connections in the database.

```
Client Side Java applet

import java.applet.*;
```

```
import java.awt.*;
import java.awt.event.*;
import java.net.*;
import vrml.external.*;
import vrml.external.field.*;
import vrml.external.exception.*;

public class Client extends Applet implements EventOutObserver, ActionListener
{
   Browser browser;      // the VRML browser
   Node root;            // root node of the loaded world
   Node proxSensor;      // first one's a proximity sensor
   Node entityGroup;     // second one's a group for holding avatars
   EventOutSFVec3f positionChanged; // eventOuts of ProximitySensor
   EventOutSFRotation orientationChanged;
   EventOutSFBool entered;
   EventInMFNode addAvatars = null; // eventIns of ObjGroup
   // other instance variables

   public void init() // initializes GUI and variables
   public void start()
   // looks for a browser
   // gets nodes from VRML model
   // adds a proximity sensor and an entity group
   // looks for position and orientation changes
   // gets avatars from the scene
   // creates a client side thread to wait for data from the server
   public void updatePosition(int, float, float, float)
   // updates clients' position in the scene
   public void updateOrientation(int, float, float, float, float)
   // updates clients' orientation in the scene
   public void updateChatBox(String) // updates messages in chatting window
   public void shutdown() // terminates client's connection
   public void callback(EventOut, double, Object) // sends information to VRML
   public void actionPerformed(ActionEvent) // sends back information
} // Client
```

The client executes two threads: TransformReceiver and ChatReceiver, which receive messages from the server, determines whether they are MOVE or ROTATION messages, and calls the relevant method in Client.

Client Side Java Thread

```
import java.net.*;
import java.io.*;
import java.util.*;

class TransformReceiver extends Thread
{
   DataInputStream in = null;
   Client muclient;
   // other instance variables

   public Receiver(DataInputStream, Client) // constructor
   public void run() // receives all clients' information from the server
} // TransformReceiver
```

3.3 Communication Interface

Every time the user navigates through the scene, the VRML browser sends position and rotation information to the server. When a second or third browser loads the same scene, it also connects to the server. From that point, movement information from one browser is sent to other browsers and is shared via the server. With this interaction between users, the VR system interface will make it possible to develop a real-time collaborative system for architectural environments via the Internet.

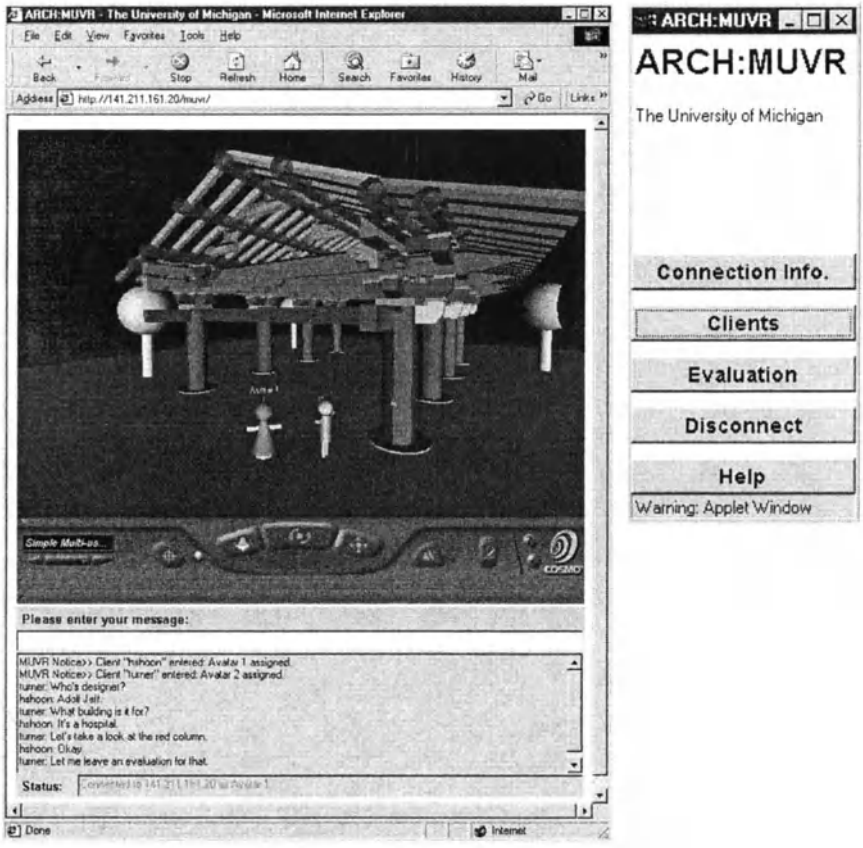


Figure 7. User Interface Supporting Collaborative Environments

The ARCH:MUVR system includes a communication interface which is a tool for sharing clients' messages, remarks, and immediate thoughts while navigating through the virtual places. All open interactions occurring are displayed in a textual chatting window identifying other clients and their actions within the scene. This communication interface is called when the

client-side applet is activated and the contents of the communication are recorded in a server-side log file.

Afterwards, all the information logged into the system can be analyzed, and attached to design updates for tested environments. The communication interface has a link to the evaluation interface which enables the clients to leave more specific comments. This feedback procedure will enable every member-user to participate collaboratively in a step-by-step design process.

3.4 Evaluation Interface

The participants in the system not only can communicate through the interface and leave their annotations, but can construct their evaluation criteria for four major characteristics for a building presentation, which is the most crucial aspect of a collaborative environment: Well-formedness, Completeness, Generality, and Efficiency (L. Khemlany and Y. Kalay).

These interactive additions for evaluation criteria are to support many different characteristics for different surroundings. They need to be stored so that other clients can participate later. ColdFusion scripts have been implemented to handle them and interacted with the system database that supports the Open Database Connectivity (ODBC) standard. Application pages are the functional parts of a ColdFusion application, including the user interface pages and forms that handle data input and format data output.

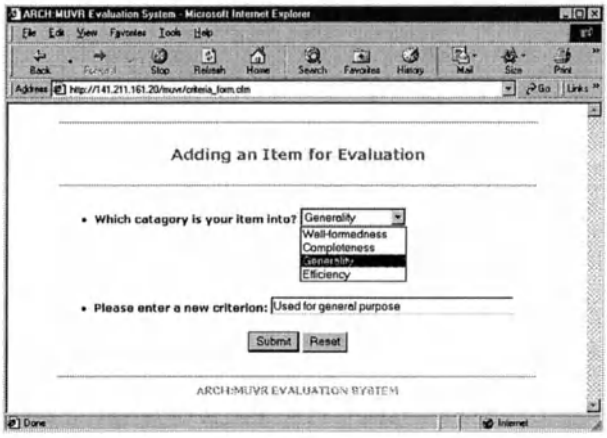


Figure 8. Creating Evaluation Criteria for the Building Presentation

The evaluation interface supports several Web page forms through server-side ColdFusion scripts, which assist to create new evaluation values, enter clients' evaluations, and save them into the database with login information. This ColdFusion access to the database and to Java

applications is possible by JDBC technology based on the ODBC. In general, ODBC enables direct connection, making two discrete systems integrated. JDBC serves as a mechanical joint between programming code objects and project database entities.

Evaluations consist of scoring evaluation criteria by selecting a radio button associated with a score value, and leaving a comment for each criterion in a text field. Scores and comments entered in the form are sent to the server-side and saved in the system database with a client ID and a date written. A later appendix from other clients can be filled out through the same interface and added immediately to the database as new records. Then, all evaluations stored will be retrieved and displayed on the result form in chronological order (*Figure 10*).

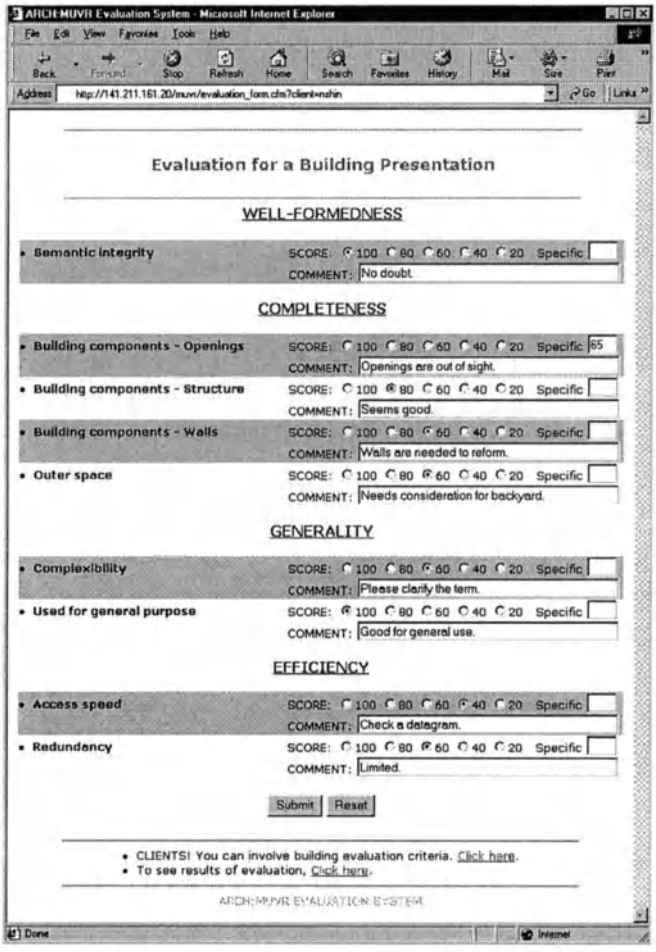


Figure 9. Evaluating for the Collaborative Environment

Scores given to each criterion are calculated by some functions, such as MAX, MIN and AVG, embedded in the Structured Query Language (SQL) and displayed on the result form. This evaluation result will allow the clients to review the building presentation and the design objects. The evaluation system is an ongoing process and, in the next version, VRML world updates followed by evaluations will be attempted.

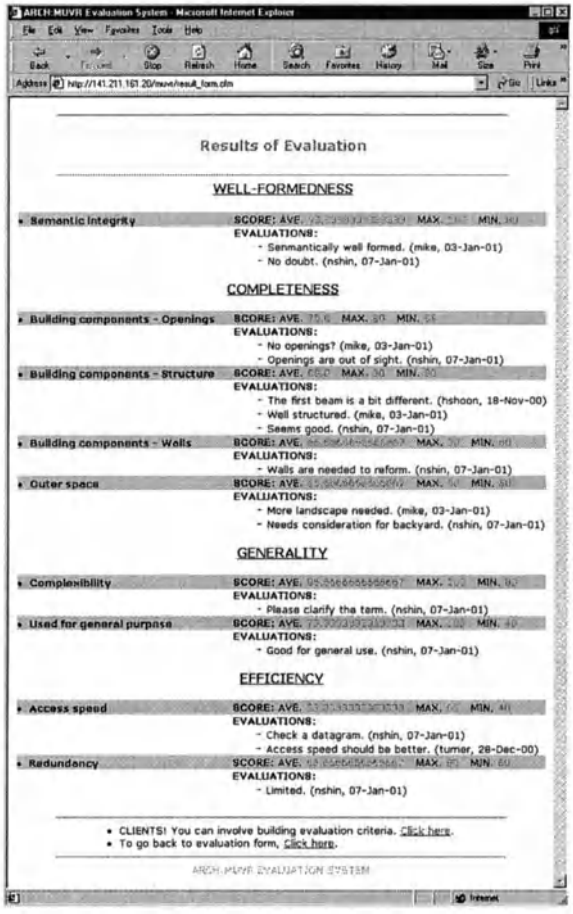


Figure 10. Results of Evaluation

4. CURRENT AND FUTURE WORK

This paper demonstrates the potential of a Web-based modeling markup language called VRML with Client-Server technology. The visualization and manipulation of an abstract model has the potential to lighten the

architects' cognitive burden in various design tasks. In future version, 3D-based direct manipulation and mapping concepts will be incorporated in the design and implementation process.

DVE offers an effective approach to collaborative work because it integrates both the communicative tools to improve collaboration and the distributed environment to elaborate information across networks.

This research also proposes an abstract model designed to provide a CAD information interface for the Internet. This abstract model will not only provide supplementary visualization tools for in-house project participants, but it will also become an interface for other remote participants who need a simplified project information-browsing tool.

With the collaborative VR system presented above, a distributed system for architectural purpose will be developed. The events dealt with the proposed distributed system could be extended by CORBA (Common Object Request Broker Architecture) event service, which allows remote calls between objects written in different programming languages and addresses dynamically updating VRML prototypes in a distributed setting.

5. REFERENCES

Bridges, A. and Charitos, D., 1997, "The Architectural Design of Virtual Environments", *Proceedings of CAAD Futures '97*, Kluwer Academic Publishers, Boston, p. 719-732.

Caneparo, L., 1997, "Shared Virtual Reality for Architectural Design", *Proceedings of CAAD Futures '97*, Kluwer Academic Publishers, Boston, p. 431-442.

Carson, J. and Clark, A., 1999, "Multicast Shared Virtual Worlds Using VRML 97", *Proceedings of VRML 99 Fourth Symposium on the Virtual Reality Modeling language*, The Association for Computing Machinery, Inc. New York, p. 133-140.

Gosling, J., Joy, B. and Steele, G., 1996, *The Java Language Specification*, Addison-Wesley Longman, Inc., Glenview.

Jung, T., Do, E., and Gross, M., 1999, "Immersive Redlining and Annotation of 3D Design Models on the Web", *Proceedings of CAAD Futures '99*, Kluwer Academic Publishers, Boston, p. 81-98.

Khemlani, L. and Kalay, Y., 1997, "An Integrated Computing Environment for Collaborative, Multi-disciplinary Building Design", *Proceedings of CAAD Futures '97*, Kluwer Academic Publishers, Boston, p. 389-416.

Lea, R., Matsuda, K. and Miyashita, K., 1996, *Java for 3D and VRML Worlds*, New Riders Publishing, Indianapolis.

Roehl, B., Couch, J., Reed-Ballreich, C., Rohaly, T., and Brown, G., 1997, *Late Night VRML 2.0 with Java*, Ziff-Davis Press, Emeryville.

Saar, K., 1999, "VIRTUS: A Collaborative Multi-user Platform", *Proceedings of VRML 99 Fourth Symposium on the Virtual Reality Modeling language*, The Association for Computing Machinery, Inc. New York, p. 141-152.

Swing, E., 2000, "Adding Immersion to Collaborative Tools", *Proceedings of Web3D - VRML 2000 Fifth Symposium on the Virtual Reality Modeling language*. The Association for Computing Machinery, Inc, New York., p. 63-68.

A Form-making Algorithm

Shape Grammar Reversed

Park, Hyoung-June and Emmanuel-George Vakaló
University of Michigan

Key words: CAAD, Design Research, Design Languages, Design Process Model, and Internet

Abstract: Under the assumption that design is a hypothetical test of building, this paper introduces a method of employing architectural knowledge through direct manipulation of geometric objects. Proposing a framework for retrieving and analysing not only what is modelled but also how it is designed, this paper demonstrates that ***designing*** can be viewed as an object of research. The paper also discusses the issues pertaining to the implementation of the aforementioned framework.

1. INTRODUCTION

Based upon the possibilities of exploring spatial design with Froebel's gifts, Stiny, in his article "Kindergarten Grammar," (Stiny, 1980) develops a visual grammar to formalise a vocabulary of building elements and a system of categories of forms in languages of designs. The languages are formed by combining or augmenting other languages of designs in terms of various language-theoretic operations such as substitution (addition and deletion,) Boolean operations (union, intersection, difference) and basic transformation functions (translation, rotation, mirroring, and scaling.) Stiny shows possible techniques of identifying a spatial relation in his grammar.

After this, a number of efforts have been made in making an application of a three-dimensional shape grammar. Piazzalunga and Fitzhorn sketch a possible way of recognising a three-dimensional shape grammar, and a framework of the shape grammar application (Piazzalunga and Fitzhorn, 1998). Agarwal, Cagan, and Constantine suggest the idea of optimising production system according to the feedback about each separate

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 453–466.
© 2001 *Kluwer Academic Publishers.*

stage of designing (Agarwal, Cagan, et al., 1998). Nevertheless, the use of shape grammar still has been difficult for a student and a designer who are more familiar with manipulating formal objects than with making a rule and applying the rule to design. It is because a rule application algorithm in shape grammar generates a low degree of freedom in design with imposing a generating a rule itself as a constraint of design.

With the review of the current rule application algorithm in shape grammar, this paper introduces an algorithm of form making based upon designer's direct manipulation of objects as a possible solution to the rule-generating problem embedded in current shape grammar application. According to this form-making algorithm, a framework of computational application is proposed in this paper.

The framework has three theoretical buttresses for its implementation. The first buttress entails ***formalisation*** that translates a designer's form-making process into a set of "form-making rules."[1] The second involves ***data abstraction*** that stores the design algorithm, which is generated during design process, into a database in the format of the "Extensible Mark-up Language (XML)" (Maruyama, Tamura, et al., 1999). This form-making information is stored when a designer derives a new design object with spatial relations between different objects. During this storing process, "Object-Relational Database Management System (ORDBMS)" (Soutou, 2000) is employed for maximising the ability of querying and accessing design knowledge in database. The design knowledge consists of the annotations of each design object and the relation between different design objects. The third entails ***communication*** that applies each stored information to design. The stored information inside the database is exchanged throughout the Internet.

The framework proposed in the paper allows a designer to manage each design object in three different types of information such as "*Artifact*,"[2] "*Building Information*,"[3] and "*Design Algorithm*."[4] Three different types of design information embedded in a design object allow the designer to analyse and modify various aspects of his or her design. Now, an individual way of designing, tacit knowledge, becomes an object of play.

[1] The form-making rules consist of basic transformation functions (translation, rotation, mirroring, and scaling), spatial relations, which are addition, deletion, and Boolean operations (union, difference, and intersection), and spatial elements such as points, lines, planes, and solids.

[2] A visualised geometric representation of a design object

[3] An architecturally categorised information of constructing a design object

[4] A set of form-making rules (schemas) established by a designer's direct manipulation of a design object

2. A SHAPE GRAMMAR

As defined by Stiny, a shape grammar is a four-tuple (S, L, R, I), in which (1) S is a finite set of shapes; (2) L is a finite set of symbols; (3) R is a finite set of shape rules; (4) I is an initial shape. Design solutions defined by a shape grammar are generated by applying the shape rules in the set R to the initial shape I and to shapes produced from I. In specific, the set of sequentially ordered rules for making a design solution is "schema." A shape rule in the set R has a normal form **A → B**, where **A** and **B** are labelled shapes in $(S, L)^+$ [5]and $(S, L)^*$ [6], respectively. In this paper, all shapes are regarded as solids in space, which are shapes in U_{33}. Stiny defines the algorithm of shape rule application in his article "Shape Rules: closure, continuity, and emergence" (Stiny, 1994) as follows;

If you have a rule for your design developing	**A → B**
Apply your rule to a shape	**C**
if a transformation of **A** is a subset of **C**	**T(A) ⊆ C**
Delete a transformation of **A** from **C**, and add a transformation of **B**	**(C - T(A)) + T(B)**
Then, you will have a new shape **C'** form **C** with a rule **A → B**	**C'**

This algorithm has been employed as a method for analysing architectural precedents. Several grammars such as "Palladian Grammar (Stiny and Mitchell, 1978)," "The Grammar of Paradise (Stiny and Mitchell, 1981), " "The Language of Prairie: Frank Lloyd Wright's Prairie Houses (Koning and Eizenberg, 1981)" showed the possibility of using shape grammar in the research of traditional buildings. However, the burden of generating a rule to apply the rule itself to design has limited the usage of shape grammar in design practice.

Without a certain rule or intention of developing a design process, the current algorithm of shape grammar cannot proceed any further design step as illustrated above. In addition, another problematic point is that most designers do not have a rule or solution to every design problem. It leads a designer to manipulate shapes for finding or generating a rule. The current algorithm does not sufficiently explain a connection between designer's manipulation and generating a rule in shape grammar. In addition, the current algorithm does not clearly explain how to introduce the interpretation of the semantic part of architectural design although it effectively represents the syntax of the design. Comparing to shape grammar explained by Stiny, Chomsky's generative grammar (Chomsky, 1978) consists of the basic

[5] $(S,L)^+$ is a set which contains all labelled shapes made up of shapes and symbols in the set S and L

[6] $(S, L)^*$ is a set which contains $(S,L)^+$ and the empty labelled shape $<S_{\varnothing}, \varnothing>$

components, which are lexicon and rewriting rules. Lexicon represents a list of ***words***. It shows that the generative grammar concerns a meaningful language, which is composed of ***words***. However, Stiny's shape grammar represents a world with geometric elements, which may be compared to ***letters or alphabets*** instead of ***words*** that have meanings. Therefore, without an introduction of meaning embedded in design object, a shape grammar may produce an ambiguity in terms of confusion not creativity.

3. A FORM-MAKING ALGORITHM

The importance of direct manipulation of object in design process has been acknowledged among architects and designers since Frank Lloyd Wright stated, in his biography, the influence of playing Froebel's gifts in kindergarten method on his design. Also, it is highlighted by a few American pragmatists such as Pierce and Dewey. Dewey describes a pattern of design action in inquiry as "the controlled or directed transformation" (Dewey, 1986). Pierce suggests that the habits of purposeful actions are the rules or patterns of solving problems in the process of inquiry (Pierce, 1966).

With the hypothesis that direct manipulation of object can be the rules of making a transient progress of design or inquiry, an algorithm for translating the manipulation into a rule during the design process is proposed below.

When you have a shape $\mathbf{A}$ to be developed	$\mathbf{A}$
Make a transformation of a shape α	$\mathbf{T}(\alpha)$
such that $\alpha \in \{ \varnothing, \ldots, \mathbf{A}, \ldots, * \}$	
Then, define a spatial relation $\otimes$ between $\mathbf{A}$ and $\mathbf{T}(\alpha)$	$\mathbf{A} \otimes \mathbf{T}(\alpha)$
such that $\otimes \in$ { addition, deletion, union, difference, intersection}	
Whenever you get $\mathbf{B}$ such that $\mathbf{B} = \mathbf{A} \otimes \mathbf{T}(\alpha)$	
The relation between A and B is stored as a rule	$\mathbf{A} \rightarrow \mathbf{B}$

Where $\mathbf{A}$, $\mathbf{B}$, and α are in U_{33} (Solids in Space) Also, the addition is $\varnothing + \alpha$ and the deletion is $\beta - \alpha = \varnothing$ such that $\varnothing$ is empty shape and $\beta \subseteq \alpha$ where α and β in U_{33}.

With the suggested algorithm, a designer can focus on his/her design without the burden of shape rule making. Whenever a design solution is achieved through substitution or Boolean operation between an initial shape and a transformed shape, the relation between the initial shape and the solution is stored as a rule in a machine. At each cycle of this algorithm, a designer is able to attach an architectural meaning to each solution. This process provides a mapping of designer's meaning to a rule in shape grammar. The mapping leads a designer to apply the stored rules for solving other design problems. At the end of design process, the whole series of

shape rules of a final design object are organised with proper meaning attached in a machine. This architectural reference mapped to a final design object allows a designer to change the part of his/her final design result not only modifying a shape/ design object but also alternating the rule assigned to the shape/ design object. Thus, a syntactic intervention of design process is possibly achieved with modifying the rules generated by designer's direct manipulation of shapes during design process.

4. MAKING AN ARCHITECTURAL REFERENCE

Based upon the proposed algorithm, a tool for making an architectural reference is introduced. The tool employs the notion of *"object"* [7] and suggests a way of understanding design as a process of making *"a meaningful order"* (Papanek, 1984) and a set of building information annotated to a designed shape in U_{33}, a solid in space.

The proposed tool regards all the components used to generate a design as a set of *object*s, which are organised in sequence of design resolution. The basic structure of *object* consists of state and behaviour. State contains geometric entities as attributes of *object*. Behaviour has basic transformation functions. In addition, the spatial relation between different shapes in U_{33} defines a step, which generates a new shape. The relation includes addition, deletion, and Boolean Operations. A step *object*, which is the design resolution, clarifies schema known as series of rules. Then, each step *object* is embedded as one of attributes of the building information of a new shape. The new shape is represented in two different aspects. The first aspect is the shape as an *object* containing geometric information. The second is as an *object* in the spatial relation with other *objects*. Therefore, with this architectural reference, a user of this tool will get geometric data of a designed shape and his/her design algorithm of deriving the shape.

4.1 Formalisation

Formalisation allows a user of this tool to define shapes in U_{33} as *objects*. When the user makes addition or deletion of a shape with instantiating prototypes or using previously defined the shape, the tool creates an *object*. The attributes of the *object*, which are geometric entities, are established either by the user or defined as default value initially by the tool. Either the tool or the designer gives corresponding label or name to the *object*.

[7] An object always has two characteristics: state and behaviour. For example, Bicycles have state (current gear, current pedal cadence, two wheels, number of gears) and behaviour (braking, accelerating, slowing down, changing gears)

However, only the tool defines the identity number of the *object*. In addition, the behaviour (basic transformation functions) of the *object* is defined by the user's direct manipulation of the shape and organised by the tool. Also, the user creates a new shape by making a spatial relation between different shapes. This process is recorded as a step *object*. The created *objects* are also stored as instances for the future usage.

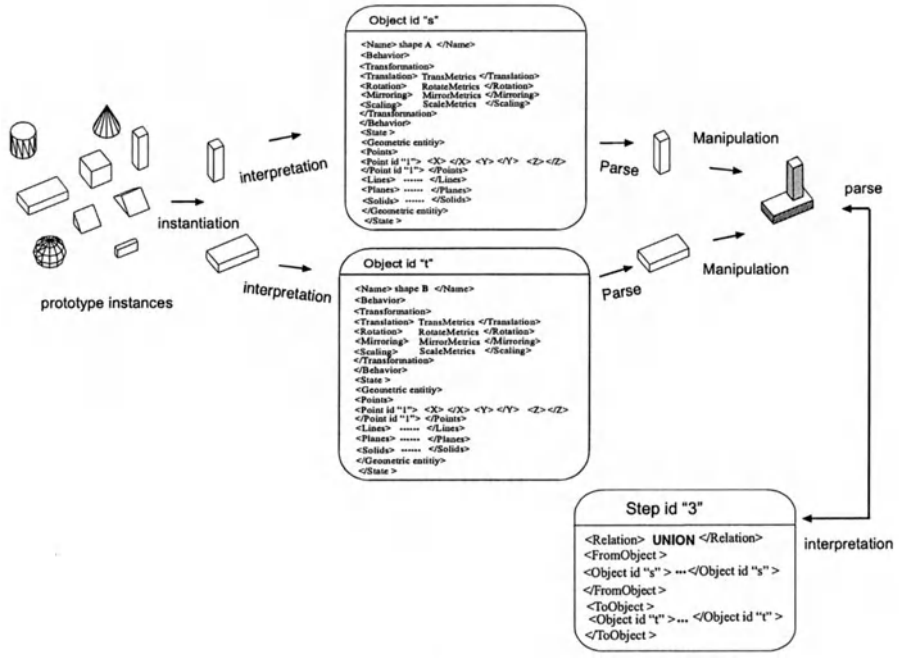


Figure 1. Formalisation

4.2 Data Abstraction

Data abstraction enables a user of the tool to isolate "how a compound *object* is used from the details of how it is constructed from more primitive *objects*. " (Abelson and Sussman, 1996) Assuming the tool provides the basic transformation functions of spatial entities, the instantiated *objects* can be assembled with various sets of formal relations defined by the user. The tool organizes each step of creating another/new *object* according to the user's assembling of shapes in U_{33}. Illustrating steps of the user's design process as schema, the tool provides an understanding of the evolution of a shape and a set of rules that generates the shape. Therefore, the sequence of making a new shape is displayed as a step *object*, and the transformation of a

shape is explained as an *object* in XML format. With the structured design information, the user of this tool is able to assemble the subset of the shape not only in constructing a compound shape within a visual state but also in syntactically modifying the process and transformations of the *object*. Figure 2 shows the data abstraction layers of the *object*, step *object*, and schema.

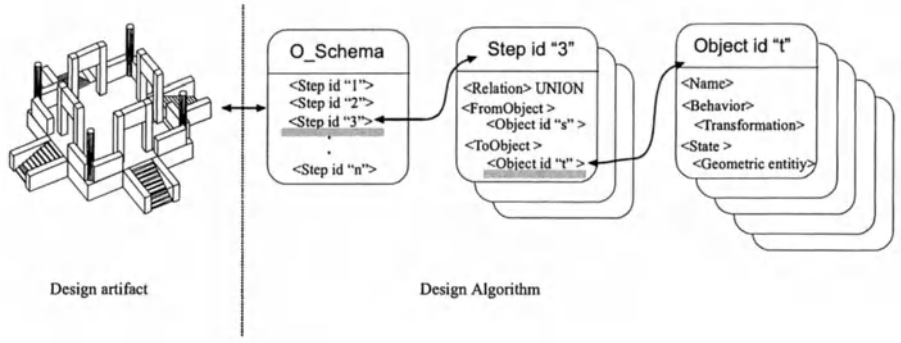


Figure 2. Data Abstraction Layers

XML is a way of structuring information in cross platform. Also, XML is syntax for establishing the formation of the hierarchical containers, which include various data type. Therefore, basically an XML document is a tree of elements in a certain order. Since XML allows a user to define the structure of each document within document-type definition (DTD), it has the advantages of describing meta-content, publishing database contents, and communicating data using a messaging format. With these advantages, XML provides a possible way to retrieve a design information as the set of programming codes instead of the format of DXF (Data eXchange File.) Thus, it helps reducing the size of memory for saving a design information. Also, XML allows a user to understand design/designing as a procedural development of structured information with manipulating shapes. The structured information is managed in ORDBMS. ORDBMS is employed for managing the storage of an organised design information. As an extension of Relation Database system (RDBMS) for affording Object-Oriented design concept, ORDBMS provides user-defined types including data structures, collection, encapsulation, inheritance, and Object Identity. Collections are a means of storing a series of data entries as a group. Encapsulation implies that data abstraction and data hiding. It also provides the hierarchy between data objects. Inheritance implies that a child of father data object can have the characters of the father object. With these features, ORDBMS allows the designer to perform complex analytical and data manipulation for searching

and retrieving various objects. Comparing to Object-Oriented Database System, ORDBMS provides frequent querying / updating access to large collection of data. (Ramakrishnan, 1998) In this proposed tool, this factor is vital since a design information is supposed to be generated whenever a new shape is created from a designer's manipulation. Figure 3 illustrates the data model of each information object managed in Oracle Designer 6.

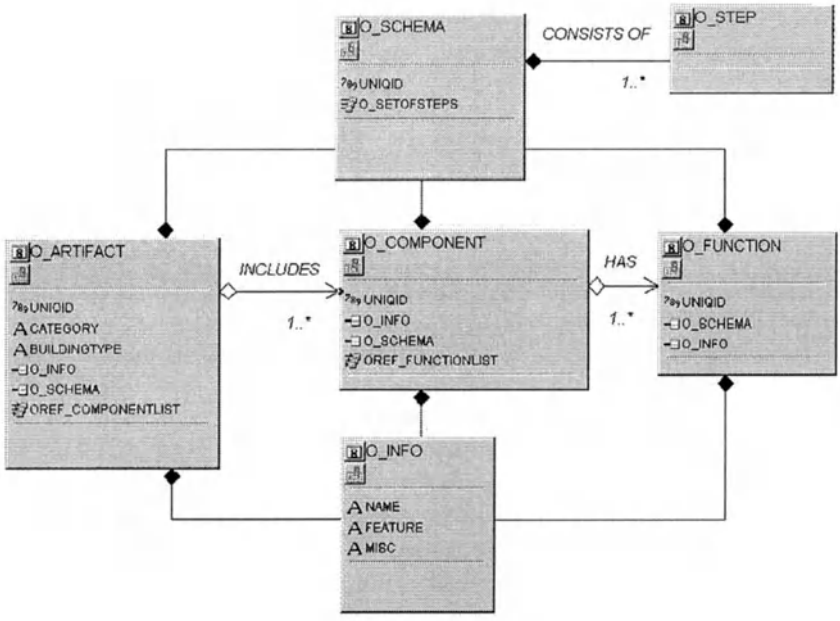


Figure 3. Data Model

4.3 Communication

To apply the proposed architectural reference within ORDBMS, establishing a network of communication is critical. It is necessary for facilitating, through the web-browser, the exchange of encapsulated information in the database with other architectural references. With the exchange, feedback and error-elimination are requested to the user of this proposed tool. According to the result of communication, the user can change the state and composition of shapes by altering the *object*, step *object* and building information parsed inside the database.

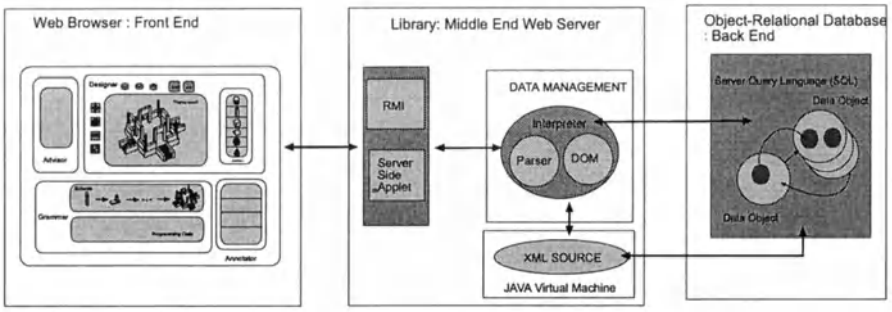


Figure 4. Communication Network

Initially **Archive** in the front end contains sets of prototype shapes as *objects*. The sets of prototype instances are pre-defined. However, in **Archive,** the number of *objects* and step *objects* increases when a new shape is constructed in **Designer** interface. According to changes occurred either in **Designer** or in **Grammar, Interpreter** in the middle end translates the shape constructed by the user into *object* and step *object* in the format of XML. In return, **Interpreter** illustrates the parsed *object* on the **Archive** panel in **Designer**, and design developing procedures on the **Schema** panel in **Grammar**. The components of the tool are illustrated in Figure 5.

The interpreted shape in XML is the subset of an *object* that makes up a final shape. It leads the user to find out how his or her final shape consists of sets of individual shapes. Also, **Interpreter** in the middle end converts each command of search conducted in **Annotator** to Server Query Language (SQL) for proper function of ORDBMS. In addition, **Interpreter** re-organises the parsed step *object* in LSP format as a programming code. In return, the programming code itself is appeared on the code panel in **Grammar**. The final shape is divided into *objects* and step *objects* in the format of XML during a user's design developing. With the help of network communication based upon "Remote Method Invocation (RMI) and Server Side Applet (SERVLET)," (Reese, 1997) the user is able to employ the basic transformation functions and spatial relations, which are transferred to the database in the back end server, in **Designer** or **Grammar**.

4.4 Implementation

4.4.1 Components

The components of the tool for making an architectural reference are **Designer**, **Grammar**, **Annotator**, **Archive**, and **Advisor**.

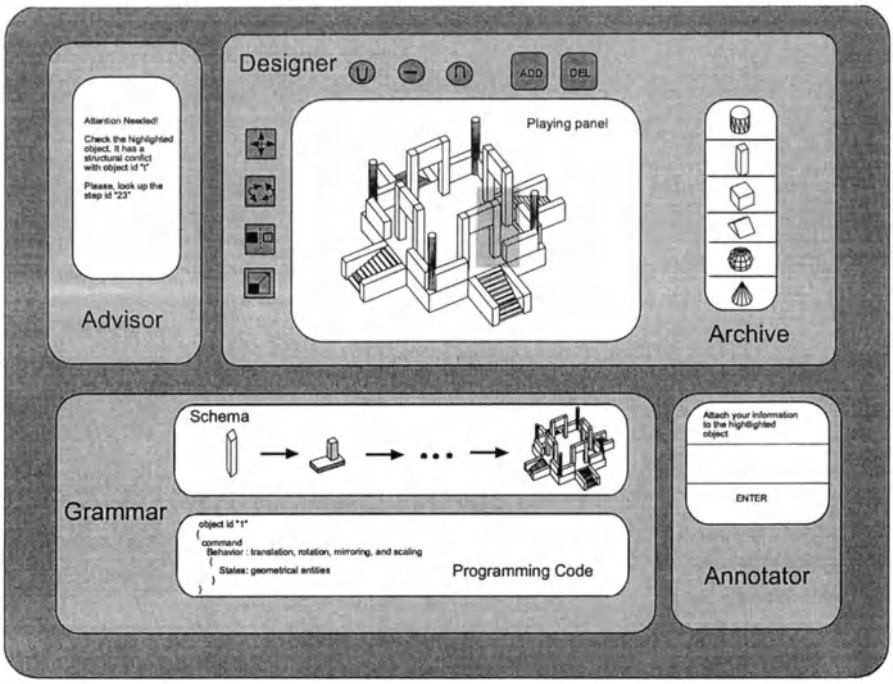


Figure 5. Use-Interface

Designer gives a user of this tool several options for developing his/her design such as basic transformations, substitution, and Boolean operations. Also, **Designer** visualises design shapes in U_{33} according to the user's direct manipulation of the shapes. In **Grammar**, the user is able to perform a syntactical intervention of design process by modifying a design algorithm, which is generated from **Designer**. The intervention is made with changing contents in **Programming Code** or altering the sequence of the evolution of shapes in **Schema**. Also, the intervention made in **Grammar** effects the state of a shape represented in **Designer**. With **Annotator**, the user can attach a building information to each shape according to his/her need. In addition, **Annotator** helps the user search and update the instances in **Archive**. The search and update method leads the user to make a systematic comparison. During design process, the category, type, function and constructional information of each shape are organised as the contents of a data object named "O_Artifact" in a database. It allows user to study his/her design with adequate information. In **Advisor** panel, possible problems embedded in each shape are displayed based upon the exchange of design information among different architectural references.

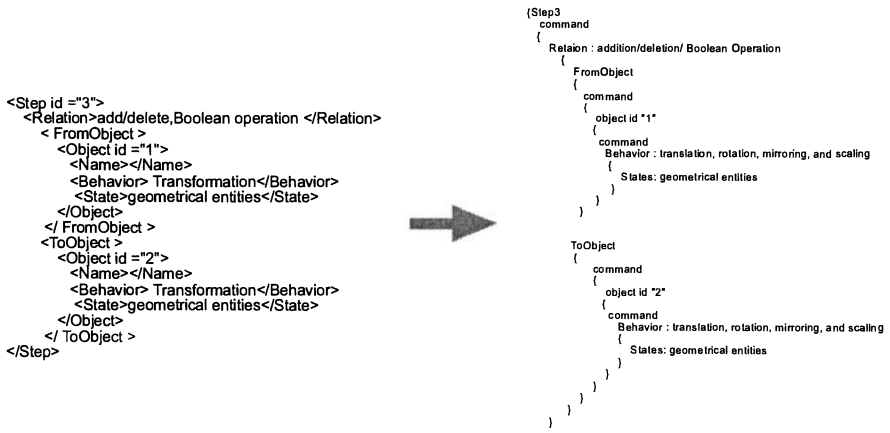
4.4.2 From Shape to Object

When a user selects and transforms a certain shape, a temporary memory is cached for storing the state of the selected shape. Comparing the state and the changed state, the transformation used for the changes is specified: A = B * T, then B^{-1} * A = T where A is a 4 x 4 matrix of the selected shape. B is a 4 x 4 matrix of an altered shape. B^{-1} is an inverse matrix of B. T is the basic transformation function matrix (Translation, Rotation, Mirroring, or Scaling). The initial states of shapes are already known since the user is supposed to begin his/her design with instantiating prototypes contained in **Archive** panel of **Designer,** which is in front end. According to information of the states and behavior of a shape, **Interpreter** in the middle end writes an *object* as a description of the shape in XML format.

4.4.3 From Object to Step Object

A step *object* is specified only when a user makes a new shape by defining a spatial relation between shapes. The spatial relation is defined by the user's direct manipulation of the shapes. And the identity number of step *object* is automatically defined in sequential order only by ORDBMS. The step *object* contains FromObjects, ToObjects, and the spatial relation. Especially, ToObject of addition is always Ø (void)

4.4.4 From Step Object to Programming Code



5. CONCLUSION

The tool for making an architectural reference has been developed as a migration of Nine Square Grid Composition (NSGC), which was designed within AutoCAD environment in AutoLisp and DCL, to a web-based application in C++ and JAVA programming language. The basic structure of NSGC is rooted from research on designer work in traditional studio class.[8] There are two most polemic points taken from the research. The first is that rational discussion between designer and instructor about a design is not possible without records of the form-making process. The second is that the necessity of an architectural reference for searching, comparing, and retrieving design artifacts during design process with a computational application.

With the proposed form-making algorithm focused on a direct manipulation of design objects and its translation to a rule, a tool for making an architectural reference shows the possibility of implementing shape grammar in constructive design practice. By introducing the concept of Object-Oriented Design, we tried to explain how individual design process could be programmed in terms of *object*s and step *object*s. Without the burden of understanding a programming language, designers create easily their own programming of what they design, and investigate their algorithm of form-making process. Also, providing a way of recording building information of each design artifact in a database, we suggested the model of an architectural reference for studying and developing the design artifact.

The further development should regard parametric transformation as one of basic transformations for affording more flexible design. In addition, for achieving more sophisticated design result, a way of combining design itself with information of other design disciplinary areas, which are history, environment, structure, urban planning and so on, in the data structure is needed.

6. ACKNOWLEDGEMENTS

In memory of his intellectual excellence and humour, I contribute this paper to my mentor and friend, the Late Emmanuel-George Vakaló.

[8] http://www-personal.umich.edu/~egvakalo/nsgc/teaching/design.htm

7. REFERENCES

Abelson and Sussman, 1996, *Structure and Interpretation of Programs*, The MIT press, Cambridge, Massachusetts, p. 79-93.

Agarwal, M., Cagan, J & Constantine, K.G.,1998, "Influencing Generative Design Through Continuous Evaluation: Associating Costs with the Coffeemaker Shape Grammar" *Artificial Intelligence for Engineering Design, Analysis and Manufacturing (AIEDAM)* Vol. 13 Number 4, p. 253-275.

Alexander, C., 1967, *Notes on the Synthesis of Form.* , Harvard University Press, Cambridge, Massachusetts.

Chen, K and Owen ,C. , 1997, "Form Language and Style Description", *Design Studies* 18, Elsevier Science Ltd, Great Britain, p.249-274.

Chomsky, N., 1978, *Syntactic structures*, The Hague : Mouton.

Dewey, J. , 1986, *Logic-The Theory of Inquiry: Volume12, The Later Works of John Dewey, 1925-1953.* , Southern Illinois University Press , Carbondale.

Koning, H. and Eizenberg. , 1981, "The Language of the Prairie: Frank Lloyd Wright's Prairie Houses ", *Environment and Planning B: Planning and Design,* Vol.8, p. 295-323.

Lulushi, A., 1998, *Inside Oracle Designer/ 2000*, Prentice Hall, Inc., New Jersey.

Maruyama, H., Tamura, K and Uramoto, N. , 1999, *XML and JAVA*, Addison- Wesley Longman Inc. , Massachusetts, p. 14-30.

Mitchell, W.J. , 1989, *The Logic of Architecture,* MIT press, Cambridge, Massachusetts.

Papanek, V., 1984, *Design for the Real World.* Van Nostrand Reinhold.

Perry, M , 1998, "Coordinating Joint Design Work: the Role of Communication and Artefacts", *Design Studies* 19, Elsevier Science Ltd, Great Britain, p. 273-288.

Piazzalunga, U. and Fitzhorn, P. I., 1998, "Note on a Three-dimensional Shape Grammar Interpreter", *Environment and Planning B: Planning and Design* Vol25, p. 11-33.

Peirce, C.S. , 1966, "How to Make Our Ideas Clear", in *Selected Writings of Charles S. Peirce(Values in a Universe of Chance)*, edited by Philip P.W, Dover Publications, New York.

Ramakrishnan, R. , 1998, *Database Management Systems* , WCB/McGraw-Hill, INC. Boston, Massachusetts, p. 614-645.

Reese, G. , 1997, *Database Programming With JDBC and JAVA* , O'Reilly & Associates, INC. Cambridge, Massachusetts, p. 139-169.

Soutou, C. , 2000, "Modeling relationships in object-relational database", *Data & Knowledge Engineering*, vol. 36, p. 79-107.

Stiny, G. and Gips, J., 1972, "Shape Grammars and the Generative Specification of Painting and Sculpture" in C V Freiman (Ed) *Information Processing 71,* Amsterdam, North-Holland, p. 1460-1465.

Stiny, G. and Mitchell, W.J., 1978, "The Palladian Grammars " , *Environment and Planning B5: Planning and Design* Vol1, p. 5-18.

Stiny, G., 1980, "Kindergarten Grammars: Designing With Froebel's Building Gifts" , *Environment and Planning B: Planning and Design* Vol7, p. 409-462.

Stiny, G. and Mitchell, W.J., 1981, "The Grammar of Paradise ", *Environment and Planning B7: Planning and Design* Vol2, p. 209-226.

Stiny, G., 1981, "A Note on the Description of Designs ", *Environment and Planning B: Planning and Design* Vol8, p. 257-267.

Stiny, G. and March, L. , 1981, "Design Machines", *Environment and Planning B: Planning and Design* Vol8, p. 245-255.

Stiny, G., 1992, "Weights", *Environment and Planning B: Planning and Design,* Vol9, p. 413-430.

Stiny, G., 1994, " Shape Rules: closure, continuity, and emergence", *Environmental Planning B: Planning and Design,* Vol21, p. 49-78.

Szkman, S , Racz J, Bochenek C, and Sriram, R. D , 2000, "A Web-based System for Design Artifact modeling", *Design Studies* 21, Elsevier Science Ltd, Great Britain, p. 145-165.

Waite, M , 1998, *Object-Oriented Design in Java*, Stephen Gilbert and BillMaCarty Corre Madera, California.

Yair, K., 1999, "Design Through Making: crafts knowledge as facilitator to collaborative new product development", *Design Studies* 20, Elsevier Science Ltd, Great Britain, p. 495-515.

Zeisel J., 1981, *Inquiry by Design*, Brooks/Cole Publishing Company, Monterely, California.

A framework for redesign using FBS models and grammar adaptation

Scott C. Chase[1] and PakSan Liew
University of Sydney

Key words: Redesign, Stylistic Change, Feature Grammars, Function-Behaviour-Structure models

Abstract: This paper describes a framework for redesign. Stylistic change in the form of rule modification is used to transform grammars to produce designs conforming to new requirements. The mechanism that enables this modification is based on the Function-Behaviour-Structure (FBS) model of design. The framework provides a formal mechanism for redesign and defines a means to generate and link structures with different behaviour and functions within the FBS model of design. Redesign of a wall illustrates this framework.

1. INTRODUCTION

Design often involves different disciplines, each forming a particular domain with its specific viewpoint on the design. For example, an architect and a structural engineer each have their own requirements and interpretations of the design. The architect may consider spatial configuration and leave the structural engineer to design the load bearing capabilities of the building elements.

An initial design created in one domain is usually subject to modification by another, as each domain has different requirements. In this context, *redesign* is considered as the process of modifying an existing design based on additional criteria from another domain. The current workable design is the starting point for the redesign process and additional requirements from dif-

[1] currently at the University of Strathclyde

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 467–477.
© 2001 *Kluwer Academic Publishers.*

ferent domains will modify this initial design to conform to additional requirements.

This paper describes a formal framework for redesign. The representational foundation for this framework is based on feature grammars (Brown, McMahon et al., 1995). The concept of stylistic changes as defined by rule modifications (Knight, 1994), serves as a basis for replacing rules used in the derivations of the original design with one that produce designs conforming to new requirements. The mechanism that enables this replacement is based on the Function-Behaviour-Structure (FBS) model of design (Gero, Tham et al., 1992). The main concepts for the redesign framework are described in the sections following.

The framework process is illustrated with the redesign of a wall, loosely based on the EDM project's descriptions of panel and core walls (Eastman, Bond et al., 1991).

2. FEATURE GRAMMARS AND KNOWLEDGE BASES

We assume here that original designs are typically produced with a CAD tool such as a solid modeller. The design process can be carried out with native geometries or predefined feature sets. The only requirement is that the resulting model be interpretable using feature sets of a specified domain, as needed. Typically, features are represented as graphs in a solid model data structure to enable graph grammars to be used for automatic feature recognition (Pinilla, Finger et al., 1989; Chuang and Henderson, 1991).

A domain-specific knowledge base is used for creating a feature grammar based on the functional requirements of the design. This grammar is used to parse the original design to create a structure based on the rules in the grammar. Each rule of this grammar has associated descriptions that provide functional or behavioural information for the features used. The parsing process extracts feature information from the initial design model, allowing reconstruction of the initial design via a feature grammar with a set of functional or behavioural descriptions.

The same knowledge base is also used to generate a set of rules that defines the underlying requirements for that particular domain. Each of these rules has a similar functional or behavioural description associated with it.

3. FBS MODEL OF DESIGN

In the FBS (Function-Behaviour-Structure) framework of design knowledge, function defines *what* the artefact does, while structure specifies the component parts and their interconnections. Behaviour specifies *how* the structure of the artefact achieves the required function and acts as a link between function and structure.

This framework of design knowledge explicates the relationships between function, behaviour and structure of an artefact and facilitates explicit reasoning among them (Gero, Tham et al., 1992). The incorporation of behaviour breaks the rigid coupling between form (structure) and function, and "enhances the range of computational models" by providing "the opportunity to explore a wider variety of solution principles without prejudice to certain artefacts allowing for innovative new solutions" (Welch and Dixon, 1992). Only behaviours that contribute to the fulfilment of functions specific to a particular viewpoint are considered.

The relationships between function, behaviour and structure are described using dependency networks. These networks show explicitly the relational, computational and qualitative knowledge among function, behaviour and structure in a schematic form (Gero, Tham et al., 1992). Graphs can be used to represent the dependency network (Qian and Gero, 1996). Each node in the network represents either a function; behaviour; structure or their characterising variables and the link across nodes describes the dependency between them. *Figure 1* illustrates a sample FBS representation for a panel wall.

Previous work using FBS paradigms has focussed on the static representation of FBS descriptions and not on their generation. As we are interested in redesign, we consider dynamic descriptions and propose a methodology for constructing FBS descriptions.

4. DESIGN DESCRIPTIONS

A formal grammar can be used to generate a design with an associated FBS description. As the geometric representation of the design is generated using the transformation rules, its FBS description is created by a function that maps designs in the language of the grammar to descriptions from a predefined set. The description function is specified by defining a description for the initial geometric state in the design grammar and a corresponding mapping function, *g(i)* for each transformation rule *i* (Stiny, 1981).

In our work design descriptions are used to model the FBS characteristics of a design. The description function creates an FBS representation of

the design as it is generated by the design grammar. This description is then used for the redesign of the original artefact based on new requirements for the FBS characteristics of the design.

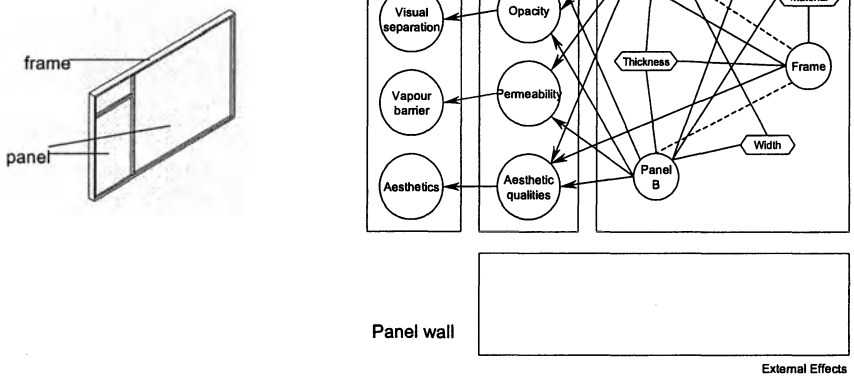


Figure 1. A panel wall and a corresponding FBS representation

5. STYLISTIC CHANGE AND GRAMMAR ADAPTATION

Terry Knight (1994) has demonstrated how systematic modification of shape grammars can encapsulate stylistic change. This occurs through addition, deletion or modification of grammar rules, often by shape replacement or modification of spatial relations. While capturing these changes in a very clear manner, there is little mention of the motivation for any transformation (admittedly a difficult thing to ascertain and decidedly outside the scope of her work).

In our work we expand the scope of these transformations to include ones based on modification of the functional, behavioural or structural characteristics of designs, as defined in their FBS descriptions. These transformations manifest themselves as rule replacements, and are motivated by specific requirements for redesign, e.g. new functional requirements.

6. REDESIGN FRAMEWORK

The redesign process is based on specific operations acting upon the functional, behavioural and structural properties of the original design within the context of additional requirements. An FBS description of the original design is first constructed and later modified. *Figure 2* illustrates the processes for this redesign framework, with explanation in the following sections.

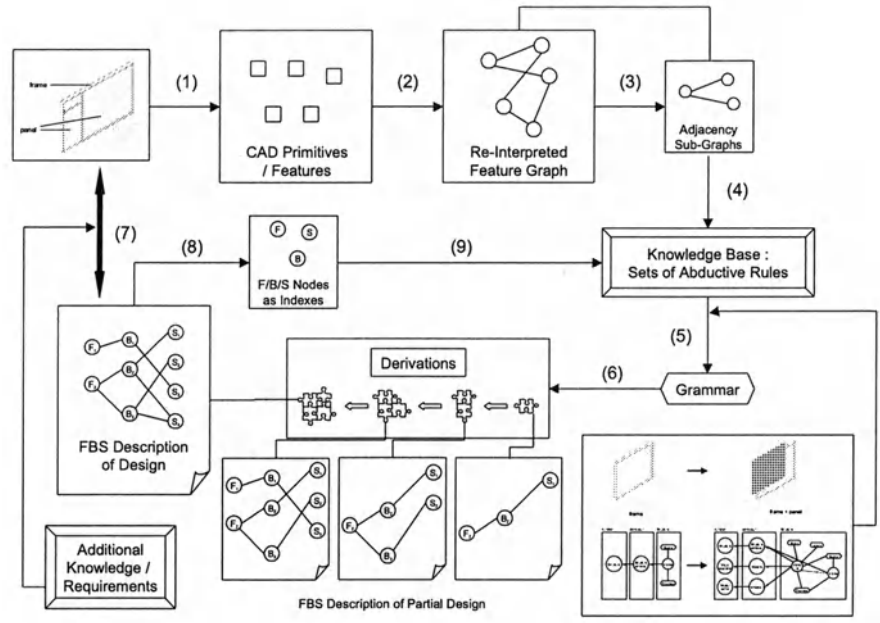


Figure 2. Construction of FBS description and the redesign process

6.1 Part I: Construction of FBS Descriptions

A design is created with a design tool such as a solid modeller by using its native geometries or predefined feature sets (1). The resulting model is reinterpreted as graphs of the required features specific to the relevant domain for redesigning (2). Typically, geometric features are represented as graphs of geometric entities in a solid model data structure and graph grammars can be used for domain-specific feature interpretation (Pinilla, Finger et al., 1989; Chuang and Henderson, 1991).

Adjacency sub-graphs of this feature graph are extracted (3) and used to search a knowledge base (4) for abductive rules that model part of the designer's knowledge (Gero and Maher, 1990). Each of these rules *R* is mod-

elled with a mapping function *g(R)* that generates its FBS description. The structure part of this description contains adjacency information between physical elements that is matched against the adjacency sub-graphs from the original design for the retrieval of relevant rules. *Figure 3* illustrates an example of one such rule.

A grammar is assembled from a set of such rules (5) and then used to construct a derivation of the original design (6). This derivation recreates the original subassembly with an associated FBS description. This description is represented as graphs that define the various functional, behavioural and structural properties of the original design as nodes and their relationships as arcs. Additional input from the designer may be required to form a more complete FBS description, as the derivation only provides a partial description (7). This additional information can be obtained from a functional analysis of the original design.

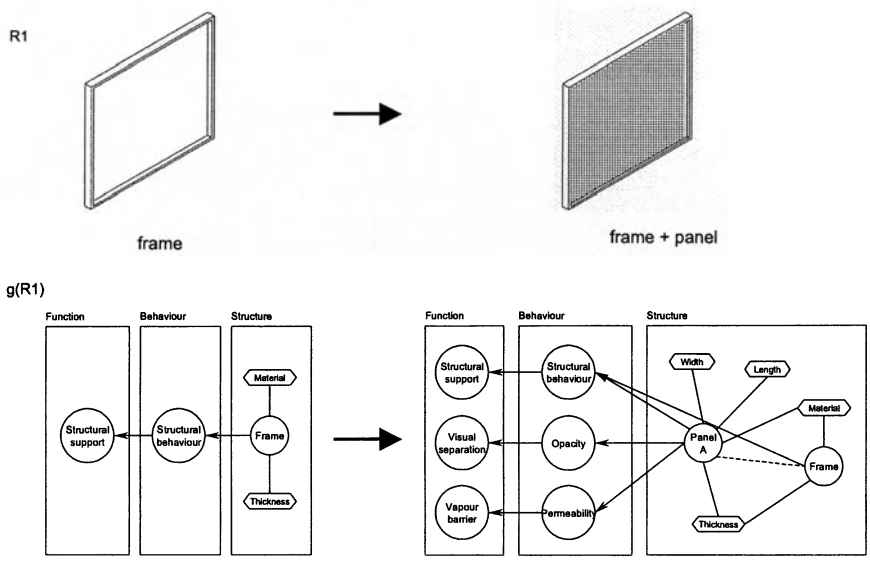


Figure 3. A sample abductive rule R1 with its associated function g(R1) to generate the FBS description of a panel wall

6.2 Part II: Redesign

To carry out redesigning, the various nodes in the FBS description of the original design are used as indexes (8) in the search for alternatives in the library of abductive rules (9). New rules are selected, based on modified or additional requirements for the functional, behavioural and structural properties of the original design. An example in the domain of Design for As-

sembly (DFA) might consider the use of alternative devices for fastening in place of bolts and nuts, based on a requirement of fewer parts. The function "fastening" would be used as an index to search for relevant rules containing the function "fastening" in their associated FBS descriptions.

The original rules in the grammar are replaced by the newly selected rules, resulting in the transformation or "adaptation" of the original grammar. A new design is regenerated using the adapted grammar and its corresponding FBS description is created. Comparing the original and modified FBS descriptions highlights potential requirements for any elements that are added or modified.

7. AN EXAMPLE: WALL REDESIGN

The following example illustrates an application of the framework described above in the context of redesigning a wall to satisfy additional performance requirements. It is loosely based on the representation of panel and core walls described in the EDM project (Eastman, Bond et al., 1991).

Here we assume that the user utilises features (wall and panel components) to design the wall assembly (*Figure 4*). Any feature recognition process can thus be ignored; the frame and panels form the features involved in the wall redesign.

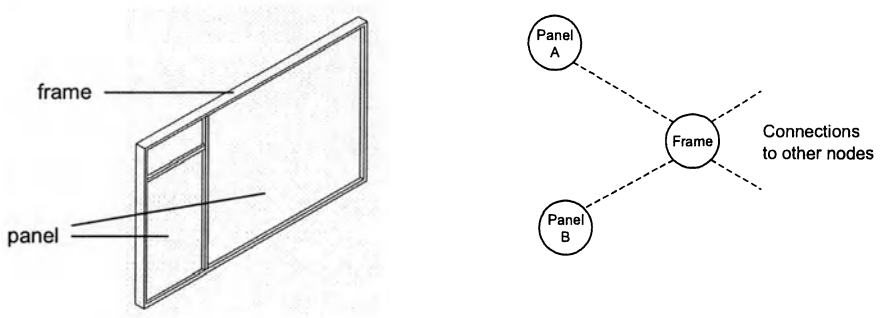


Figure 4. Panel wall and its adjacency graph

The adjacency between these features can be obtained from the CAD database of the original design. This adjacency sub-graph is used to search for rules in the knowledge base that have the same adjacency information in their FBS descriptions (e.g., match the panel-frame adjacency sub-graph on the right side of g(R1), *Figure 3* with a sub-graph of the one in *Figure 4*).

A grammar can be assembled from a set of such rules and then used to construct the derivations of the original wall assembly. This derivation generates the original wall with an associated FBS description. The description for the subassembly is illustrated in *Figure 1*. Note that the panel insertion rule provides only part of the FBS description. The function and behaviour dealing with aesthetic properties are obtained from the designer's knowledge base. Other such functions and behaviours could be added via a rigorous functional analysis, e.g. increased rigidity for the structural behaviour.

For redesign in the context of this example, the assumption is made that the acoustic and thermal properties of the proposed wall now become relevant, having previously not been considered in the FBS representation for the panel wall. These functions and behavioural requirements are added to the FBS description of the proposed wall (8) and the knowledge base is searched again for rules that contain these nodes (9). Rules such as those that construct core walls (R2, *Figure 5* and *Figure 6*) can be used to revise the grammar by replacing the analogous rules for panel walls. By replacing the rule invocations of R1 in the original derivation (6) with those of R2 and R3 (adding surfaces to core walls, not illustrated), a new core wall design is generated that satisfies the additional thermal and acoustical requirements. The original grammar has now been adapted to produce new designs according to a new set of requirements.

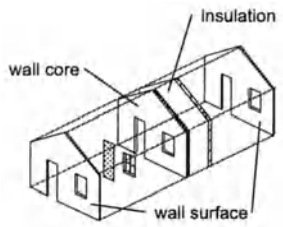


Figure 5. Core wall

The new design for the wall assembly is constructed from the derivations of the adapted grammar and its corresponding FBS description is generated. An understanding of the differences in the functional, behavioural and structural aspects of the original and modified designs can be obtained through a comparison of their FBS descriptions (*Figure 7*). Note that due to the additional functional requirements and the measured behaviours associated with them, the effect of external variables such as climate becomes relevant.

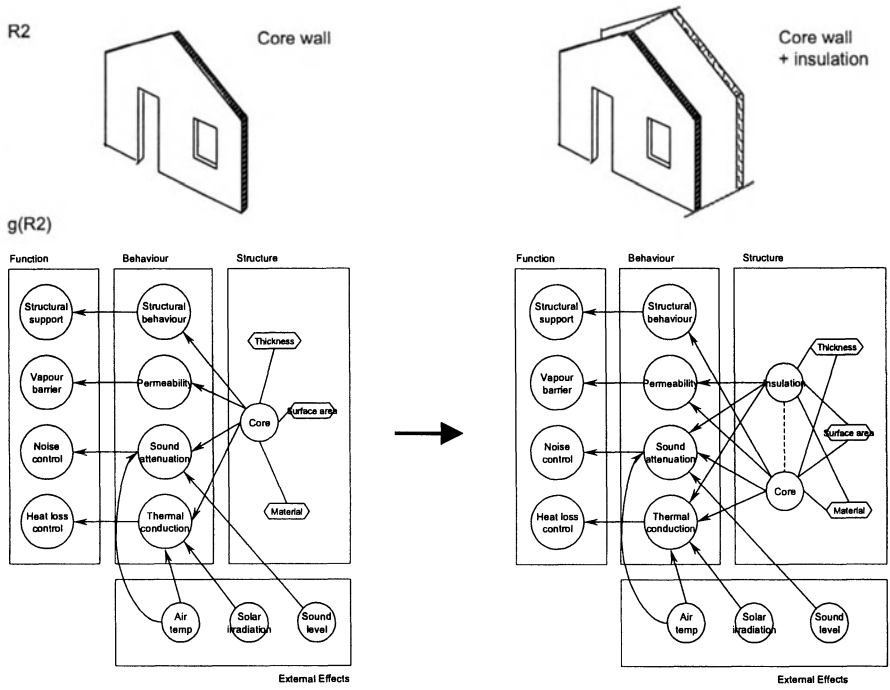


Figure 6. Rule R2 for core wall construction

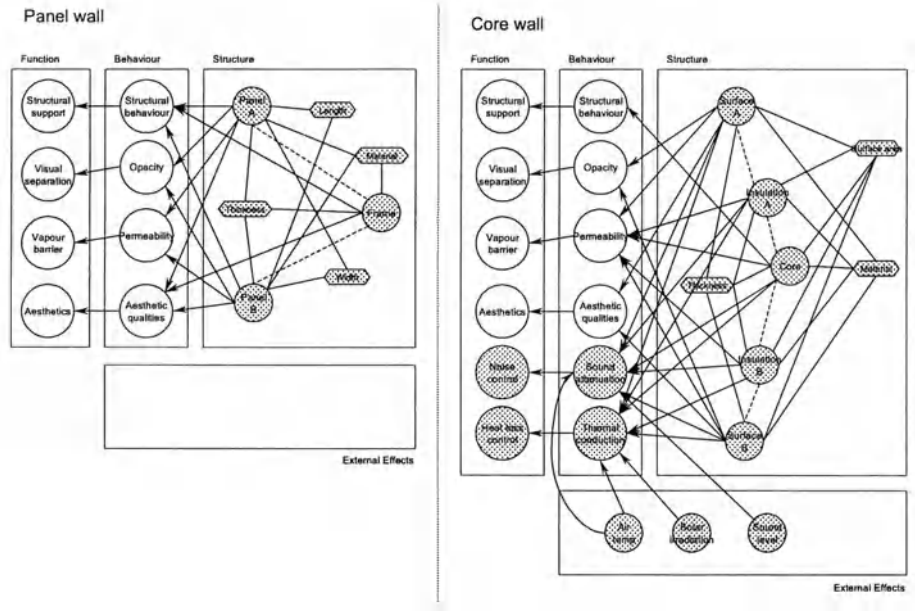


Figure 7. Comparison of the panel and core wall FBS descriptions. Differences are shaded.

In this example the introduction of new functional requirements (thermal and acoustic) resulted in the complete replacement of the original structure. Other possibilities for redesign could result in less drastic modification of the FBS network or a simple modification of the acceptable range for behavioural or structure variables. We also note that the introduction of new functions initiated the redesign process. In other situations, such as Design for Assembly, the requirement for fewer parts can lead to a partial modification of structure, but introducing new behaviours (e.g., the replacement of a nut and bolt fastener with a snap-fit adds the behaviour of elastic distortion).

8. FUTURE WORK

In this paper we have proposed a framework for redesign by grammar modification through rule replacement. Further work, including testing of this framework, is required. Additionally, other areas worthy of further investigation include:

- Use of the framework for more complex examples than the one illustrated here. How well will this methodology scale?
- Control of rule selection and invocation, especially in ambiguous, non-deterministic cases.
- Exploration of how prototype libraries can be used in redesign. For example, other possible paradigms used for redesign not examined here include combination, analogy and case-based reasoning.
- Formalisation of design principles by graph operations on FBS descriptions. One such domain could be Design for Manufacturing Assembly (DFMA), in which guiding principles include reduction of number of parts, combination of parts and simplification of designs.

9. SUMMARY AND CONCLUSIONS

In this paper we described a methodology for formalising redesign based on functional, behavioural and structure requirements. We proposed a process for the generation of FBS descriptions of designs based on adjacency graphs of their components and knowledge bases of component behaviour. By replacing the rules used to generate a design (adapting the grammar), new designs are created that meet new requirements. This framework has the potential to support the formalisation of engineering design methodologies such as DFMA.

10. ACKNOWLEDGEMENTS

This research was partially funded by the Australian Research Council Small Grant Scheme.

11. REFERENCES

Brown, K. N., C. A. McMahon and J. H. Sims Williams, 1995, "Features, aka The Semantics of a Formal Language of Manufacturing", *Research in Engineering Design* 7(3), p. 151-172.

Chuang, S.-H. and M. R. Henderson, 1991, "Compound Feature Recognition by Web Grammar Parsing", *Research in Engineering Design* 2(3), p. 147-158.

Eastman, C. M., A. H. Bond and S. C. Chase, 1991, "Application and Evaluation of an Engineering Data Model", *Research in Engineering Design* 2, p. 185-207.

Gero, J. S. and M. L. Maher, 1990, "Theoretical requirements for creative design by analogy", in: P. Fitzhorn (eds.), *First International Workshop on Formal Methods in Engineering Design, Manufacturing and Assembly, Colorado Springs, 15-17 January 1990*, Colorado State University, Fort Collins, Colo., p. 19-27.

Gero, J. S., K. W. Tham and H. S. Lee, 1992, "Behaviour: A Link Between Function and Structure in Design", in: D. C. Brown, M. Waldron and H. Yoshikawa (eds.), *Intelligent Computer Aided Design*, North-Holland, Amsterdam, p. 193-220.

Knight, T. W., 1994, *Transformations in Design: A Formal Approach to Stylistic Change and Innovation in the Visual Arts*, Cambridge University Press, Cambridge, England.

Pinilla, J. M., S. Finger and F. B. Prinz, 1989, "Shape Feature Description and Recognition Using an Augmented Topology Graph Grammar", in: (eds.), *Preprints, NSF Engineering Design Research Conference, College of Engineering, University of Massachusetts*, p. 285-300.

Qian, L. and J. S. Gero, 1996, "Function-behaviour-structure paths and their role in analogy-based design", *Artificial Intelligence for Engineering Design, Analysis and Manufacturing* 10, p. 289-312.

Stiny, G., 1981, "A note on the description of designs", *Environment and Planning B* 8, p. 257-267.

Welch, R. V. and J. R. Dixon, 1992, "Representing Function, Behavior and Structure During Conceptual Design", in: D. L. Taylor and L. A. Stauffer (eds.), *Design Theory and Methodology - DTM '92*, American Society of Mechanical Engineers, New York, p. 11-18.

Building Services Standard Solutions

Variational Generation of Plant Room Layouts

B. Medjdoub[1], P. Richens[1] and N. Barnard[2]
[1]*The Martin Centre, University of Cambridge, UK*
[2] *Oscar Faber, St Albans, Hertfordshire, UK*

Key words: constraints, layout configuration, topological solutions, optimisation, interactivity, plant room, pipe routing, 3D solution.

Abstract: Object-based CAD programming is used to take advantage of standardisation to handle the schematic design, sizing, layout and (potentially) pipe-routing for LPHW (Low Pressure Hot Water) plant rooms in buildings. From a simple specification of the plant room geometry, and the heating load in kw, our software proceeds through a number of steps. First the standard number and size of modular boilers, pumps etc. is determined from the heat load. Then a compatible optimising 3D variational solution is generated, using Constraint Logic Programming. Our approach is highly interactive. Modifying the topology of the solution is done directly through the graphic interface, e.g. modifying a boiler position is done by dragging; the system automatically updates the 3D model including the pipe-routing while maintaining all the constraints, and hence the validity of the design.

1. INTRODUCTION

If the design of any complex artefact is suitably restricted by adhering to a library of predefined components and assembly details, it becomes possible to automate a great deal of the design process. In the case of heating and ventilation plants there are in any case good value-engineering reasons to use standard components and details.

Currently, designers solve these problems "by hand". Starting from the heating load in kW, a schematic solution is defined. The engineers proceed to equipment selection, then its location, followed by pipe routing governed by objective requirements (minimum surface area, minimum pipe length, minimum bend number...). In summary plant room physical design amounts to *layout configuration* followed by *pipe routing*.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 479–493.
© 2001 *Kluwer Academic Publishers.*

Computer support for layout configuration has been studied for more than twenty years. Studies include techniques such as mathematical programming (Mitchell et al., 1976), expert systems (André, 1986; Flemming, 1988), genetic evolution (Jo and Gero, 1998; Gero and Kazakov, 1998; Rosenman, 1996) and constraint satisfaction (Baykan and Fox, 1991; Aggoun and Beldiceanu, 1992; Charman, 1994). These approaches typically enumerate all placement solutions. Similar solutions, differing only in the precise positioning of an element on the modular grid, are considered as two different *geometrical solutions*. Clearly, in preliminary design, it is wasteful to distinguish between geometrically close solutions, as this generates a high number of solutions (typically several thousands or millions) which cannot be distinguished in their global aspect by the designer. A recent approach based on constraint satisfaction has been presented by Medjdoub and Yannou (Medjdoub and Yannou, 2000) where the topology and the geometry are separated. This brings great flexibility in constraint utilisation since the constraint definition is separated from the resolution algorithms, and deals with high-order combinatorial problems.

Another disadvantage of most of the aforementioned approaches to space planning is that they attempt full automation of the process.

Our approach to plant room design combines automation and interactivity. From a simple specification of the plant room geometry (an orthogonal polygon with known obstructions, openings and external walls), and the heating load in kW, our software proceeds through a number of steps. First the number and size of standard modular boilers, pumps etc is determined from the heat load. Then a compatible optimising 3D variational solution is generated, using Constraint Logic Programming and particularly the arc-consistency[1] (Smith, 1995) on integers. To do this we firstly enumerate a satisfactory topological solution, and then refine it to form a compatible geometrical solution. The final step generates pipe routes, using branch and bound optimisation to minimise the length of pipes and the number of bends. Heuristics are used to restrict the search to promising areas of the solution space. In this NP-complete problem, dynamic variable ordering (dvo) heuristics can have a profound effect on the performance of backtracking search algorithms (Bacchus et al., 1995; Gent et al., 1996;

1 If there is a binary constraint C_{ij} between the variables x_i and x_j then the arc $(x_i; x_j)$ is arc consistent if for every value $a \in D_i$ (D_i is the Domain of x_i), there is a value $b \in D_j$ such that the assignments $x_i=a$ and $x_j=b$ satisfy the constraint C_{ij}. Any value $a \in D_i$ for which this is not true, i.e. no such value b exists, can safely be removed from D_i, since it cannot be part of any consistent solution: removing all such values makes the arc $(x_i; x_j)$ arc consistent. Note that we have only checked the values of x_i; there may still be values in the domain of x_j which could be removed if we reverse the operation and make the arc $(x_j; x_i)$ arc consistent.

Smith et al., 1998). Tsang et al. (Tsang et al., 1995) show that there does not appear to be a universally best algorithm, and that certain algorithms may be preferred under certain circumstances.

The solution obtained can be modified or improved by the designer, when we provide further contribution by concentrating on interactivity. Modifying the topology of the solution or the geometry of the plant room is done directly through the graphic interface, e.g. modifying a boiler position is done by dragging; the system automatically updates the 3D model including the pipe-routing while maintaining all the constraints.

We describe the plant room design problem in section 2. The object model of the plant room and the plant room design rules are presented in sections 3 and 4. The enumeration algorithm is described in section 5. In section 6 the interactivity of the system is presented. Before concluding, the implementation and benchmarking are presented in sections 7.

2. PROBLEM DESCRIPTION

The selection of systems for the plant room is based on its function, e.g. heating, cooling, air handling, electrical etc. The selection of equipment for each system will then be a standard solution defining the number of plant items and the interconnection configuration required to perform the function (Barnard, 1999). These solutions are typically manifested in the form of schematics (see *Figure 1*). Sizing will generally require data in the form of loads, together with rules. The data may be calculated externally, generated on the basis of a rule-of-thumb or supplied from another plant item. For example, boilers will require a heat load for sizing that may initially be estimated using a rule-of-thumb based on the building area, to be replaced by a calculated value as the design evolves. When the number of plant items to perform the function is defined, two steps remain to reach the final solution: item location and pipe routing.

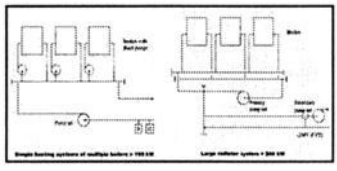


Figure 1. Example of schematic solution of plant room

Traditionally, the location of equipment is approached from two extremes. One extreme is to devise a solution where the general arrangement of the (main) equipment is predetermined on the basis of a fairly rigid set of

rules. This is appropriate for new buildings where there may be a degree of flexibility concerning the size and configuration of the plant room. The other extreme would be to allow the plant to be placed randomly within the space available. This would allow it to fit into plant rooms of various sizes. A number of possible configurations could be produced and the best selected according to a number of judgement criteria, e.g. spatial requirement, cost etc. *Figure 2* shows some preferred layouts of modular boilers, based on general location rules:

- The boiler room is usually in a separate space (must be separate from chillers).
- Each piece of equipment requires a minimum maintenance space around it.
- All equipment is on a plinth 100mm high.
- Access routes are provided through plant areas: 2000mm is ideal, 1500mm is the minimum requirement.
- Double doors leading to the plant room are required.
- A single door for emergency escape must be located opposite to the main door.
- Boiler location is side by side
- Boilers back against a wall.

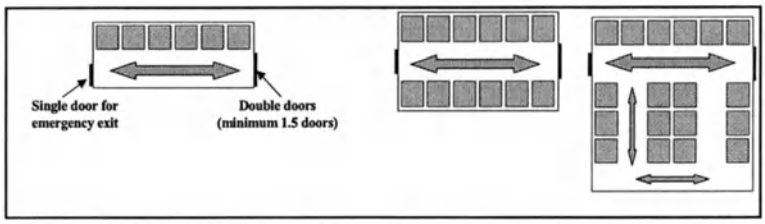


Figure 2. Favourite layout configuration of plant room

As soon as the plant room layout is defined we proceed to pipe routing. The priority is for horizontal orientation minimising the number of bends and the pipe length.

3. OBJECT MODEL OF THE PLANT ROOM

The plant room object model holds three main classes of objects representing the plant room geometry, equipment and pipework. Each defined class is characterised by attributes and class constraints.

3.1 Plant room geometry

The plant room geometry can be an orthogonal polygon with known obstructions, openings and external walls. We can also provide columns (see *Figure 3*). The object structure of the plant room geometry includes three classes: Space, Door and Column. Each class is characterised by a reference point (x, y, z), a length l, a width w and a height h. In *Figure 3* the plant room is an L-shape with two doors and one column. The L-shape corresponds to: $space_1$–$space_2$. The reference points, l, w and h, are integer-constrained variables. We use an *arc-consistency on integers* constraint programming technique which explains the need for a *distance increment*; but this is not too restrictive as the scale used is the millimetre.

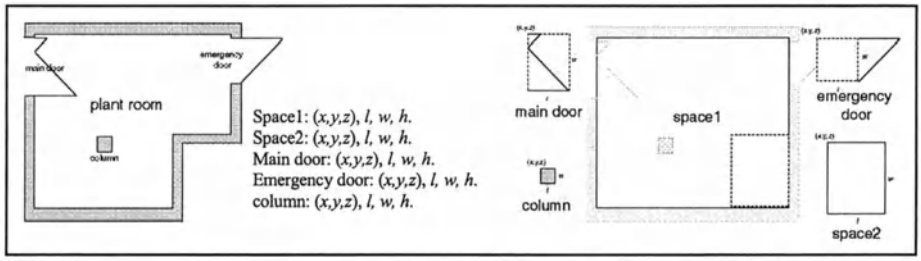


Figure 3. Plant room geometrical representation

3.2 Equipment

The equipment is described in four sub-classes: Boiler, Pump, Pressurisation unit and Control panel. Each class is characterised by a reference point (x_1, y_1, z_1), a length l, a width w, a height h, and an orientation *ori*. The reference point, l, w and h are integer-constrained variables. The orientation attribute is a constrained discrete variable defined over the domain $\{0°, 90°, 180°, 270°\}$, the length, the width and the height are fixed variables. As is indicated in *Figure 4*, each piece of equipment requires a minimum space for the maintenance.

A class constraint is defined to ensure the four possible configurations corresponding to the four possible orientations. This constraint generates a choice point leading to four constrained sub-problems, these are shown in *Figure 5*.

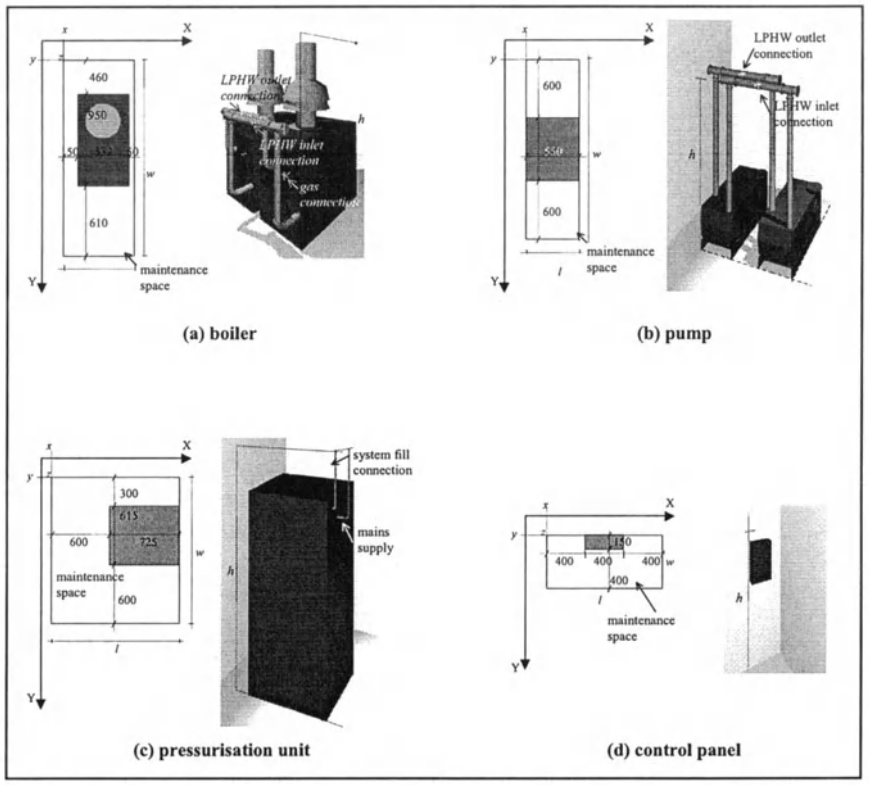


Figure 4. Geometrical representation of the plant room equipment

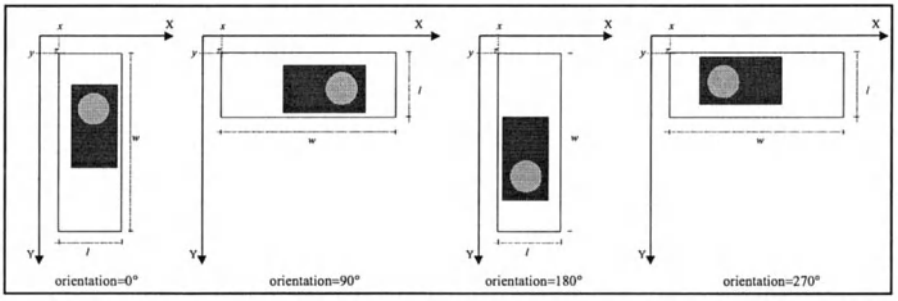


Figure 5. The four possible orientations of the boiler

3.3 Pipework

The Pipe class is defined by a set of points and a radius *r* (see *Figure 6*). Each pair of successive points defines a segment of the pipe (i.e. (P_1,P_2), (P_2,P_3) and (P_3,P_4) are the three segments of the pipe shown in *Figure 6*). An

orientation *ori* is associated with each segment. All the attributes are integer-constrained variables except the orientation which is a constrained discrete variable defined over the domain $\{x°, y°, z°\}$. A class constraint is defined to ensure that two successive segments have different orientation.

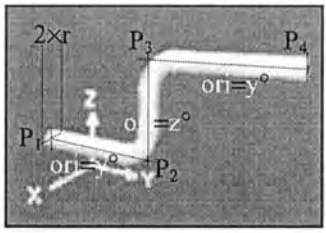


Figure 6. Geometrical representation of a pipe with a set of four points (P_1, P_2, P_3, P_4) corresponding to two bends

4. PLANT ROOM DESIGN RULES

The plant room design rules are expressed by constraints. These constraints are applied implicitly by the system and can be divided into two categories: topological and dimensional. Topological constraints define continuity and neighbourhood in a building while dimensional constraints define distances and angles (Boudon et al. 1972).

4.1 Dimensional constraints

Dimensional constraints assign a minimal or a maximal value to the object constrained variables. This constraint is expressed by equality or inequality, i.e. boiler.l=532.

4.2 Topological constraints

Topological constraints comprise:

- Inclusion constraint, i.e. all the objects are inside the plant room.
- Non-overlapping constraint between the plant room components and pipework.
- Adjacency constraint, i.e. between the control panel and the walls.

4.2.1 Inclusion in the plant room

This simple constraint consists of four conjunctive inequalities over x_1, x_1, z_1, y_2, y_2, z_2 in order to be inside of the current plant room. In the case of L-shape or T-shape (see *Figure 3*) equipment and pipework are inside space$_1$ and do not overlap space$_2$.

4.2.2 Non-overlapping

A non-overlapping constraint expresses the fact that two elements cannot overlap each other; it is automatically applied between all pairs of elements. *Figure 7* shows the position permitted for $e_2.x_2$, $e_2.y_2$ and $e_2.z_2$ ($e_2.z_2$ represents the constrained variable z_2 of element e_2) by the non-overlapping constraint between two elements e_1 and e_2. The non-overlapping constraint introduces a new non-overlapping variable with six values {E,W,N,S,A,U}. This variable divides the space surroundings into six parts (see *Figure 7*) but not symmetrically. Indeed, we observe that the N and S values give more solutions than the E and W values. This asymmetry is made to avoid any redundant solution. It is the instantiation of these non-overlapping variables and the adjacency variables which, if proven consistent, gives a topological solution.

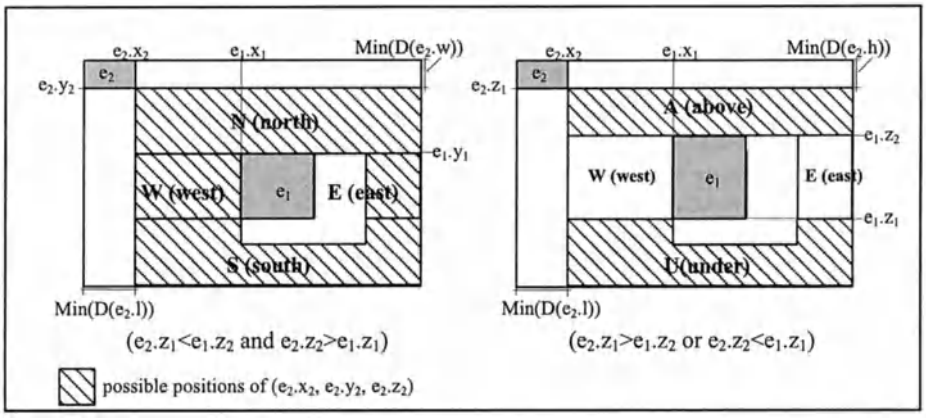


Figure 7. General non-overlapping constraint

The non-overlapping constraints between plant room equipment introduce variables with four values {E,W,N,S} as they are all on the same floor. Between the pipework and the equipment the variables generated have five values {E,W,N,S,A}, as the pipework can be located above the equipment.

4.2.3 Adjacency

The adjacency constraint is applied between the control panel and the plant room walls.

The application of this constraint creates a new discrete constrained variable named *adjacency variable* defined over the domain {E, W, N, S}, standing for east, west, north and south wall. In fact, each adjacency constraint and its consequent adjacency variable introduce a choice point in the tree search (see *Figure 8*). The adjacent constraint is a "dæmon" constraint for which an instantiation of the adjacency variable triggers a propagation and consequently a domain reduction thanks to the arc-consistency technique. During enumeration, corner solutions are only enumerated once.

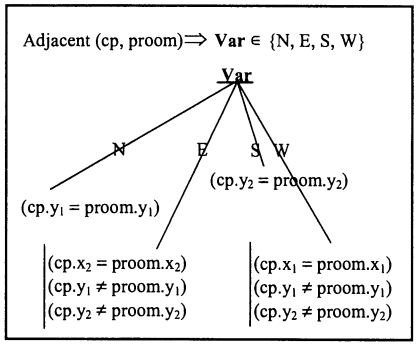


Figure 8. Adjacency between the control panel and the plant room walls

In our application, the following disjunctive form is supported: e_1 (*adjacent#1*) e_2 *OR* e_1 (*adjacent#2*) e_3 *with* $e_2 \neq e_3$. It is applied in case of L-shape or T-shape plant room. In figure 2 the control panel is constrained to be adjacent to space$_1$ walls or to space$_2$ walls.

5. ENUMERATION ALGORITHM

To generate the solution we follow two steps. First, we generate the plant room layout which determines the equipment location. Then, we proceed to pipe routing minimising the length of pipes and the number of bends.

5.1 Plant room layout generation

In this step our objective is to enumerate the first satisfactory topological solutions, and then refine it to form a 3D compatible geometrical solution.

The topological solution is based on Medjdoub and Yannou definition (Medjdoub and Yannou 2000):

Each space layout Constraint Satisfaction Problem (CSP) where the $n(n-1)/2$ (n being the number of items) non-overlapping variables and adjacency variables are instantiated and which remains geometrically consistent (i.e. for which at least one geometrical solution exists) is a topological solution.

Contrary to Medjdoub's approach, we enumerate just the first satisfactory topological solution. Dynamic variable ordering (*dvo*) heuristics are used to restrict the search to promising areas of the space tree. These heuristics are not only used for a better performance of the backtracking search algorithms, but also to lead to a better topology solution if it exists. Thus, the designer can select from the user interface (by choosing between vertical or horizontal layout as indicated in *Figure 2*) which topology the search algorithm will investigate first.

To refine the topological solution to a 3D geometrical solution we instantiate the geometrical parameters of the plant room equipment. We use the classical *dvo* heuristic for ordering the geometrical variables, thus the priority is given to the variable with the smallest domain.

5.2 Pipe routing

The pipe routing is the last step of the enumeration. As soon as we have obtained a consistent 3D geometrical solution, we determine from our user interface the wall through which the LPHW mains pass. The algorithm proceeds in three steps. First the *bends number* variables are instantiated, the smallest values being chosen first. Then the *non-overlapping* variables between pipes and pipes/equipment are instantiated. Finally, we instantiate the pipe reference points minimising the length of each pipe using the "branch and bound" algorithm (Carpaneto et al., 1995). This is the most common approach used in constraint programming to find the optimal solution. First, we create a constrained variable representing the objective function and find an initial solution, then we introduce a new constraint that the value of the objective variable must be better than in the initial solution. We repeatedly solve the new problem and tighten the constraint on the objective variable until the problem becomes insoluble: the last solution

found is then the optimal solution (Smith, 1995). *Figure 9* shows three different solutions of two pipes with different z_1 and z_2.

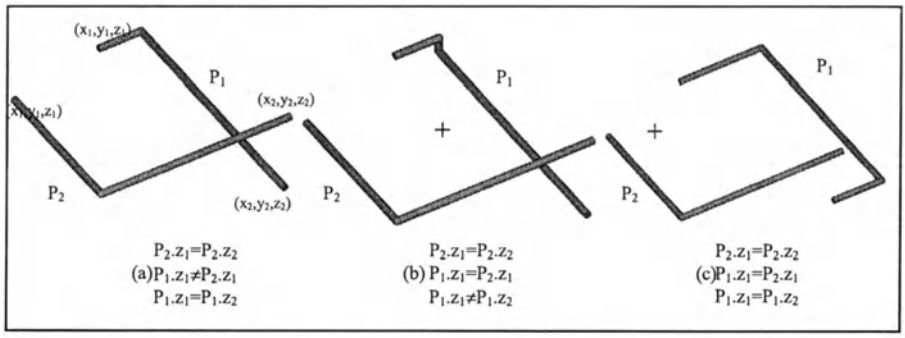


Figure 9. Three different pipework solutions

6. INTERACTIVITY

Our approach is based on interactivity. As an example we are going to generate and modify a plant room of 1GW. The first step is to generate the first satisfactory solution (see *Figure 10b*) from the defined plant room geometry (see *Figure 10a*). In this example, the heuristic of a vertical layout has been selected. The second step is to generate pipework; to do this we select the wall where the LPHW and the flue pipe pass through and run the search algorithm by clicking on a button. As indicated in Figure 1c the LPHW pipes pass through the north wall and the flue pipe passes through the west wall. After obtaining a complete 3D solution it is possible to make any modification. To modify the pipes passing through walls, we just have to select them and drag them around the walls. The flue pipe is dragged from its initial position (see *Figure 10c*) to a new one as indicated in *Figure 10d.* The same search algorithm of pipe routing is used with fixed values for all the pipes except the flue pipe.

Other features are offered by the system: plant room geometry modification and topology modification. Through the user interface, it is possible to modify the shape or size of the plant room and to add, move or delete columns and doors. From the examples shown in *Figure 10d*, by simply dragging the plant room geometry, the system generates a new solution corresponding to the new updated plant room geometry. Thus it is very easy to minimise the plant room area while maintaining all the constraints, i.e. the solution shown in *Figure 10e* corresponds to the minimum plant room area for 1GW. If the designer would like to explore

other topologies, he/she has simply to select the component and drag it to the desired position. The system generates a new solution with the desired new topology (see *Figure 10f*). If no solution exists, the previous one is maintained.

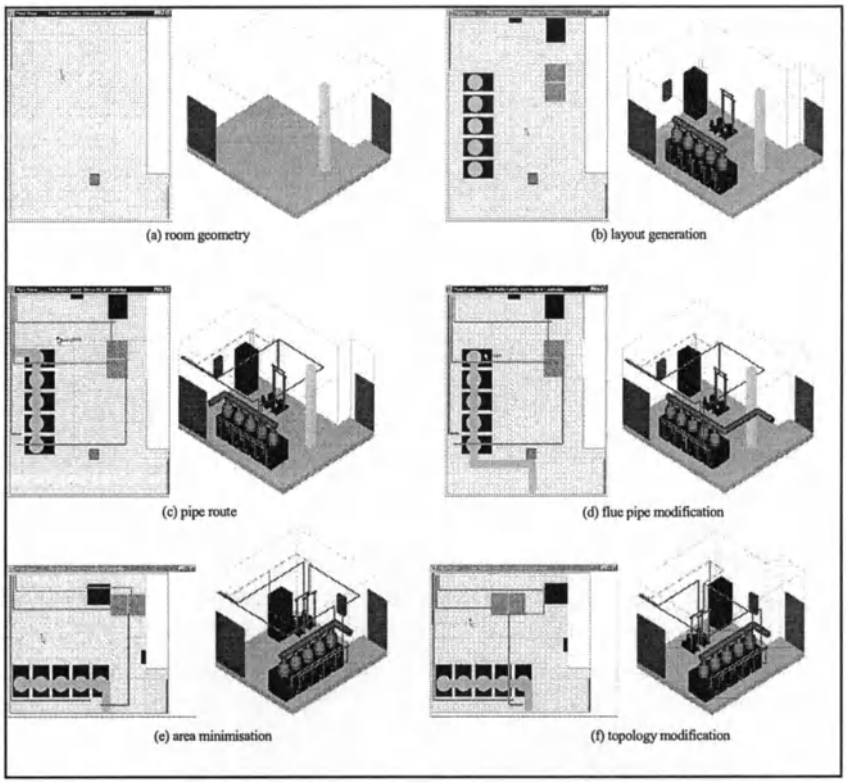


Figure 10 (a,b,c,d,e,f). 1GW plant room solution and its modification

7. IMPLEMENTATION AND BENCHMARKING

This application is developed in JMDL (*Java Modeling Language*) as embedded in MicroStation/J, and ECLiPSe[2] the constraint logic programming system. We use JNI[3] (*Java Native Interface*) as the programming interface between ECLiPSe and JMDL. We use MicroStation/J[4] to hold the object model and for 3D rendering.

The solutions generated by our prototype have been tested against

2 ECLiPSe is a trademark of IC-PARC at Imperial College, London.
3 JNI is a trademark of Sun Microsystems, Inc.
4 Microstation/J is a trademark of Bentley Systems.

conventional solutions in a benchmarking exercise (Hejab et al., 1993; Burberry, 1986). Advantages has been underlined and some advice has been suggested for future development.
The main advantages:
- the interactive layout modification.
- the interactive plant room area minimisation has been particularly appreciated.
- the 3D solution.

The main suggestions for future development:
- improving the pipe routing and more precisely the interactivity of pipe modification. The designer should be able to modify interactively the position of bends inside the plant room.
- to solve large models over 2GW. Our approach is limited to middle-size problems (50 objects between pieces of equipment and pipes).

8. CONCLUSION

The approach presented in this paper shows an interactive system for plant room design, simple to use with high-level modification of plant room layout topology, pipework position and plant room space geometry. Several aspects were discussed: the constraints and the plant room item structure, the algorithm of solution enumeration and the interaction via the user interface.

Compared to previous approaches we have here a compromise between full automation and interactivity. This balance gives to the designer full control while assisting him to solve complex problems automatically.

ACKNOWLEDGEMENTS

This project is funded by EPSRC[5] and DETR[6] under the LINK[7] Meeting Client Needs through Standardisation programme in UK. There are more than 10 partners including Oscar Faber, BSRIA, Hamworthy[2], Pullen Pumps, Bentley Systems, Taywood Engineering Ltd, The SCI, Sainsburys, Sheppard Robson, Carrier, Woods, Waterloo, Delta, Biddles, CADAC, IES Facet The programming was carried out at the Martin Centre CADLAB (University of

5 The Engineering and Physical Sciences Research Council
6 The Department of the Environment, Transport and the Regions
7 The LINK scheme is the UK Government's principal mechanism for promoting partnership in pre-competitive research between industry and the research base.

Cambridge Department of Architecture), which is supported by Informatix Inc of Tokyo.

REFERENCES

Mitchell W.J., J.P. Steadman and R.S. Liggett, 1976, "Synthesis and Optimization of a Small Rectangular Floor Plans", *Environment and Planning B*, (3): 37-70, 1976.

Andre J.M., 1986, "Vers un système d'aide intelligent pour l'aménagement spatial: CADOO", Colloque International d'IA de Marseille.

Flemming, U., 1988, "A generative expert system for the design of building layouts", Artificial Intelligence in Engineering: Design, Ed. Elsiever, New-York.

Jo J.H. and J.S. Gero, 1998, "Space Layout Planning Using an Evolutionary Approach", *Artificial Intelligence in Engineering*, 12(3), p. 149-162.

Gero, J.S. and V.A. Kasakov, 1998, "Evolving design genes in space layout planning problems", *Artificial Intelligence in Engineering*, 12(3), p. 163-176.

Rosenman M.A., 1996, "The Generation of Form Using Evolutionary Approach". AI in Design'96, J.S. Gero and F. Sudweeks (eds), Kluwer Academic, Netherlands, p. 643-662.

Baykan C., and M. Fox, 1991, "Constraint Satisfaction Techniques for Spatial Planning", *Intelligent CAD Systems III*, Practical Experience and Evaluation.

Aggoun A., and Beldiceanu N., 1992, "Extending CHIP in Order to Solve Complex Scheduling and Placement Problems". Journées françaises de la programmation logique, Marseille.

Charman Ph., 1994, "Une approche basée sur les contraintes pour la conception préliminaire des plans de sol". CERMICS-INRIA, France.

Medjdoub B. and B. Yannou, 2000, "Separating topology and geometry in space planning", *Computer Aided Design*, 32(1), p. 39-61.

Smith, B.M., 1995, A Tutorial on Constraint Programming. Report 95.14, School of Computer studies, University of Leeds, UK.

Bacchus, F. and van Run, P., 1995, Dynamic variable ordering in {CSPs}. In Montanari, U. and Rossi, F., editors, Principles and Practice of Constraint Programming, Lecture Notes in Computer Science, Cassis. Springer Verlag, p. 258-275.

Gent, I. P. et al., 1996, An empirical study of dynamic variable ordering heuristics for the constraint satisfaction problem. In Freuder editor, Principles and Practice of Constraint Programming, Lecture Notes in Computer Science, Berlin, Heidelberg, New York NY. Springer Verlag, p. 179-193.

Smith, B.M. and Grant, S.A., 1998, Trying harder to fail first. In Prade, H., editor, European Conference on Artificial Intelligence (ECAI), UK. John Wiley & Sons, p. 249-253.

Tsang, E.P.K. Borrett, J.E. and Kwan, 1995, A.C.M. An attempt to map the performance of a range of algorithm and heuristic combinations. In Hybrid Problems, Hybrid Solutions, pages. IOS Press. Proceedings of AISB, p. 203-216.

Barnard, N., 1999, "Building Services Standard Solutions: Rules and Data", Oscar Faber report R16998, St Albans, Hertfordshire, England.

Boudon Ph., et al., 1972, La geometrie chez l'architecte : Etude exploratoire, UP-architecture Nancy, France.

Carpaneto, Dell'amico & Toth, 1995, "A Branch-and-bound Algorithm for Large Scale Asymmetric Travelling Salesman Problems," *ACM Transactions on Mathematical Software* 21, p. 410-415.

Hejab, M., and C. Parsloe, 1993, "Space and Weight Allowances for Building Services Plant", The Building Services Research and Information Association, technical note TN 9/92. Bracknell, berkshire, England.

Burberry P., 1986, "Space for services - 5 Distribution and sizing", Architects' Journal, 5 March 1986, p. 32-36.

(Re)presentation of architectural analyses

Two prototype applications

Bige Tunçer, Rudi Stouffs, and Sevil Sariyildiz
Delft University of Technology

Key words: Representations, Architectural analyses, Information structures

Abstract: We present a methodology for decomposing documents by content and integrating these into a rich information structure. This implies both expanding the document structure, replacing document entities by detailed substructures, and augmenting the structure's relatedness with content information. This paper focuses on some of the representational issues involved in the process of interpreting, breaking up, and relating documents. We describe a prototype application as a tool for building up, storing, and presenting architectural analyses in an educational setting implemented using XML, discuss a similar prototype application to be implemented using *sorts*, and compare these two different methodologies.

1. INTRODUCTION

In education, as in architectural history, theory, and design, complete and thorough analyses of architectural bodies or objects are indispensable. While practitioners commonly draw from a body of design experience of their own to support their current design, students must necessarily rely on the examples of success and failure from other architects. In the past, such precedent-based learning was implicit in the master-apprentice relationship customary in the educational system. Nowadays academics no longer have the possibility to maintain an extensive design practice and students, instead, draw upon a diverse set of precedents from various prominent architects. Thus, the study of important historical precedents or designs plays an important role in design instruction and in the students' design processes (Akin, Cumming, et al., 1997). In this study, students may benefit from a collaboration with peers, by selecting each a different aspect of a same

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 495–505.
© 2001 *Kluwer Academic Publishers.*

building, or a different building with respect to the same aspect. By integrating the respective results into a common, extensible, library, students can gain from other results in comparisons and relationships between different aspects or buildings. The complexity of such integration is best supported through a computer medium.

The Web offers many examples of architectural analyses on a wide variety of subjects, with varying degrees of sophistication (Tunçer and Stouffs, 1999). Commonly, such an architectural analysis consists of a collection of abstractions, each reflecting on a different aspect such as function, acoustics, structure, and organizational relationships (Schmitt, 1993). The abstractions may be described in different formats such as drawings, diagrams, models, pictures, and textual information, and individually contained in different documents. These documents can then be categorized and hyperlinked within an organizational structure in order to support navigation through the information space. More sophisticated examples rely on a database for storage and management of the information entities, and offer a more complex categorization of the documents and their relationships. While these studies present effective ways of accessing and browsing information, it is precluded within these analyses to distinguish and relate different components within the abstractions or documents. The result is an information structure, as defined by the abstractions and the relationships between them, that is rather sparse. If enabled, instead, the decomposition of abstractions would offer a richer information structure presenting new ways of accessing, viewing, and interpreting this information.

We propose a methodology for decomposing documents by content and integrating these into a tight structure. This implies both expanding the document structure, replacing document entities by detailed substructures, and augmenting the structure's relatedness with content information. The relationships between the resulting components make the abstractions inherently related by content. This paper focuses on some of the representational issues involved in the process of interpreting, breaking up, and relating abstractions. We describe a prototype application as a tool for building up, storing, and presenting architectural analyses in an educational setting implemented using XML, discuss a similar prototype application to be implemented using *sorts*, and compare these two different methodologies.

We illustrate the potentials of the representational framework with the representation of a number of abstractions belonging to a body of built architecture, specifically, Ottoman mosques. We have selected three mosques by the same architect, Sinan (1490-1588), that present three different typologies of classical Ottoman architecture in their spatial and structural characteristics (figure 1). These are Şehzade (İstanbul), Süleymaniye (İstanbul), and Selimiye (Edirne).

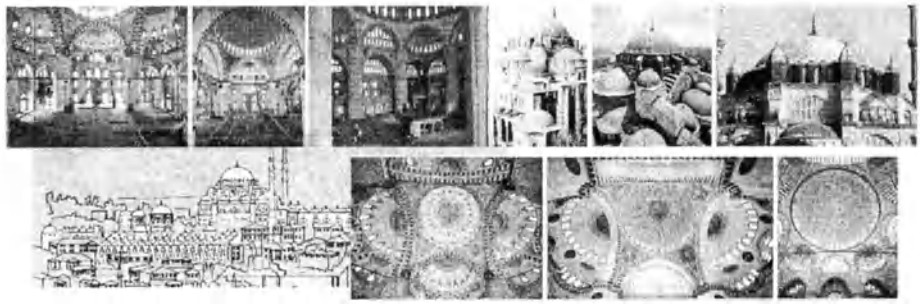


Figure 1. Sets of images from the mosques Şehzade, Süleymaniye, and Selimiye. Interior space, dome structure as seen from the outside, silhouette, and central dome(s). Images from Egli (1997), Stierlin (1998, 1985), and Erzen (1996).

2. DECOMPOSITION BY CONTENT

When using a common syntax to re-represent various abstractions, these can be interpreted and be broken up into components, components within and between abstractions can be related, and these relationships added to the representation. The result is an integrated structure of components and relationships, represented in a uniform way. Such a tightly related structure offers new possibilities for accessing, viewing, and interpreting this information. First, it allows one to access specific information directly instead of requiring a traversal of the abstraction hierarchy. Individual components can be reached and retrieved more quickly when provided with more relationships. Second, components can be considered from a different point of view. The location of a component in the structure is no longer only defined by its place in the abstraction hierarchy. Instead, components provide direct access to other related components, forming a part of the first component's view. Third and most importantly, one can access the information structure from alternative views to those that are expressed by the individual abstractions. New compositions of components and relationships offer new interpretations of the structure and generate views not inherent in the structure as created by the original abstractions. Such interpretations can lead to new abstractions.

The input to the proposed system consists of a number of abstractions. Although the approach we propose is completely flexible because it does not impose any fixed frame of reference, only a common syntax for the representation, we do offer a semantic guideline in the form of a hierarchy of types. A type can be defined as a conceptual object that represents the characteristics from a group of similar objects (Tunçer and Stouffs, 2000). A type can also be considered as an aspect of a building, such as its location in the urban fabric, or its importance in the social context of the time it was

built. The selected type hierarchy (figure 2) provides the information structure with the semantics of how the components within and between the abstractions are related. Each abstraction component is assigned at least one type and components relate through shared types and the relationships in the type hierarchy.

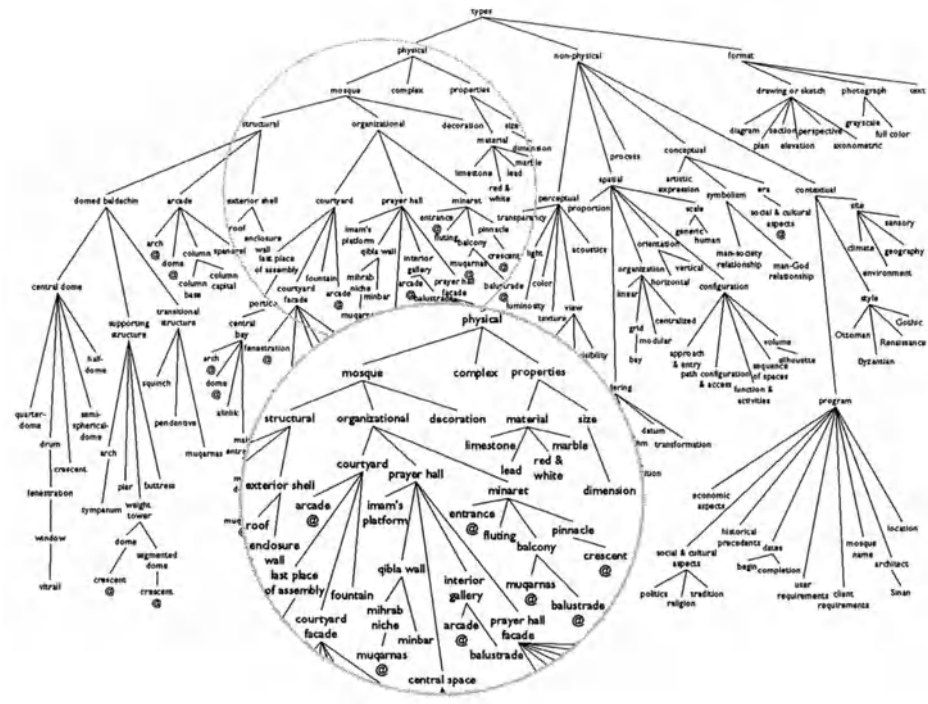


Figure 2. The type hierarchy defined for the case study. It splits into three branches at the top level: physical, non-physical, and format.

The output of the system should be an integrated structure of components and relationships. In between, a number of steps need to be traversed: abstractions are to be broken up into their components, these components within and between abstractions related.

3. XML PROTOTYPE

XML (eXtensible Markup Language) is particularly suited for the purpose of decomposing abstractions in the form of text or images and integrating them into a single structure. The strength of XML for our purpose is its ability to represent information structures: how various pieces of information relate to one another, in much the same way as a database

might. Once a structure is agreed upon, existing documents can easily be converted to XML. Using tools for scanning texts and images and recognizing keywords, concepts or patterns, such conversions can be automated.

3.1 Structure

The input to the application is a number of image and text abstractions. We are particularly concerned with texts and images in this application as these lack any strong inherent structure. Both composed of symbols from a relatively small vocabulary, i.e., characters and pixels, in simple and basic one- and two-dimensional patterns, they are represented in a similar structure and can be operated on in a similar way: divided into smaller parts and the parts organized in a hierarchical structure.

The system is composed of two main hierarchies: types and components. The grammar of XML, i.e., the DTD (Document Type Definition), specifies the structure of both hierarchies in the system: their elements, their nesting and additional properties, and their attributes (figure 3). Both hierarchies are recursively defined.

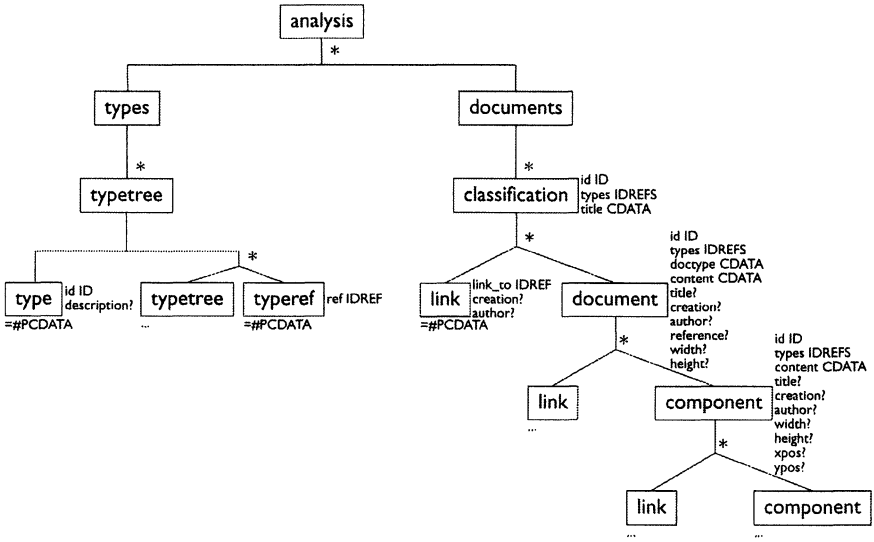


Figure 3. The grammar of the XML structures, the types and components hierarchies.

By distinguishing the components, some relationships within the abstractions have already been created. However, these relationships are purely syntactical. Semantic relationships are added through the hierarchy of types and the assignment of these types to the components. This type

hierarchy may be incorporated from an external framework or specifically defined corresponding to the subject of the analysis. This may require the hierarchy to be constructed across the viewpoints of different groups or users. The structure is defined in XML by using the type name as the tag, and by nesting the elements according to the hierarchy. Each type is additionally identified by an ID, which is used for linking to components.

The different abstractions are decomposed into their constituent entities defining the hierarchy of components. The abstractions in the form of images are broken up into sub-images by determining the important components, in correspondence to the types, and by cutting them up using an image processing application. The abstractions in the form of text are immediately structured in XML. Each component is identified by an ID, and the component hierarchy is defined by using the ID as the index, and by nesting the elements. Types are assigned to components by their ID's.

In this organization, relationships defined by the abstraction hierarchy initially relate the components. Additionally, components that share the same type are implicitly related. The type hierarchy further relates components, these relationships are derived from the nesting in the types hierarchy. Finally, explicit relationships between components can be specified as references to the component ID's. These are transferred to the XML structure as IDREFS tags.

The resulting XML structure forms a flexible source for further manipulation and traversal. Components can be flexibly categorized and grouped according to their relationships and attributes, offering various views of the information structure. Views can be traversed and linked using both explicit and implicit relationships. The XML documents are visualized through related developments such as XSL and XSLT, also using XPointer and XLink.

3.2 Interface

The system is Web-based and allows the abstractions to be broken up into components through an intuitive interface. The images are decomposed by selecting rectangular areas from the image, selecting a set of keywords from the type hierarchy, and attaching these to the image component. The texts are decomposed by selecting a piece of text and attaching keywords to it. Image recognition mechanisms for images, and keyword or concept recognition mechanisms for texts could be used to present the user with suggestions about the relevant components.

The interface allows the user to view both the type and document hierarchies and their relationships in an intuitive way. These views include both in-world and out-world views (Papanikolaou and Tunçer, 1999). An in-

world view presents a component (or type) together with its immediate neighbours within the hierarchy, and displays all other components that share a type with it (figure 4a). The in-world view allows one to browse the structure, interpret the relationships, and as such lead to interesting out-world views. While the types serve for the most part as the binding elements in the structure providing the relationships between the components, when traversing the information structure, the content as available in these components is the most important entity. As such, while the component's type, and its location in the type hierarchy, may be presented as properties of the component, the relationships are specified primarily as component-to-component relationships. This does not only ensure that the links are presented as shortly as possible, tightening the information structure, but also shifts the focus onto the content, rather than the structure that surrounds it. Types further serve a role as index to the information structure. Access to the analysis is provided through the collection of abstractions and from the types hierarchy.

In addition to the different in-world views, structural maps can provide visual feedback to the users on their traversals and selected views by presenting the location of the currently viewed node within the hierarchy. Such maps can be developed using SVG, X3D, and Java in relation to XML. An out-world view is presented as a clickable map that offers an overview of the entire type hierarchy in relationship to the related documents (figure 4b).

The presented approach provides the students with a simple interface and mechanisms for the presentation of an analysis of design precedents, and possibly their own designs. The system is designed in a way that the project grows as users add abstractions from different buildings, even from their own designs. Since all the information is integrated within a single environment, students will benefit from the different studies collected in the analysis, and can draw new conclusions across studies and presentations, including their peers'.

4. SORTS

From a representational point of view, the components and relationships recognized within an abstraction can be said to form a language (Tunçer and Stouffs, 1999), with the vocabulary of the language dependent on the representational format of the abstraction. When abstractions are collated into a single information structure, this structure defines a meta-language that is a composition of the languages of the original abstractions. Consequently, new abstractions can be considered as defined by new languages that form part of this meta-language (figure 5). Slicing the

structure for a new abstraction, then, relies on the specification of a corresponding vocabulary. According to this selected vocabulary, components and relationships will be included into the section, or excluded from it. The resulting structure defines the new abstraction. However, any ability to define new abstractions should not be conceived as the reduction of a rich structure into simpler abstractions once again. Instead, these new abstractions constitute the result of queries to the structure that are unrestricted by the original composition into abstractions.

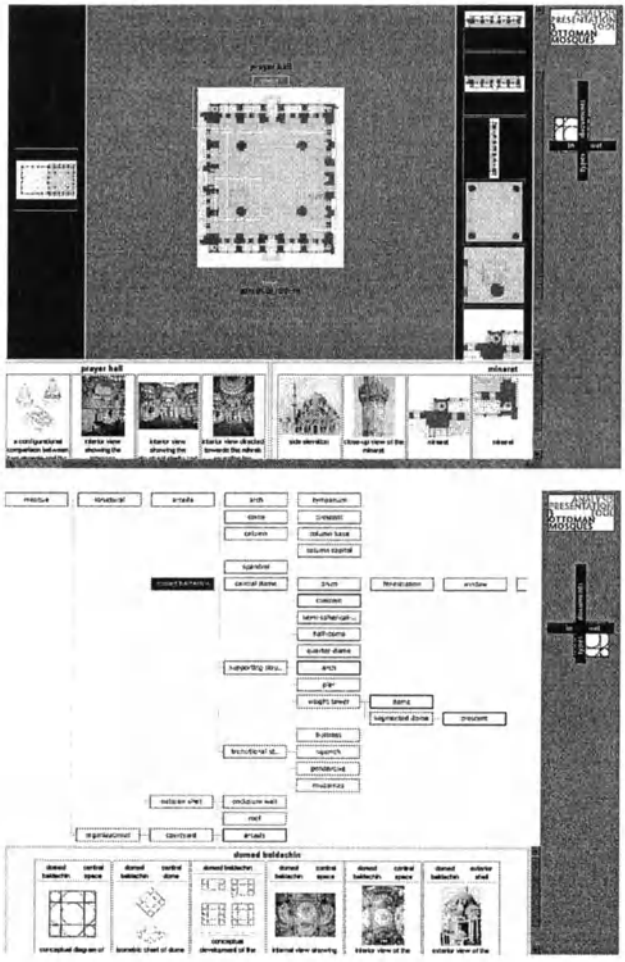


Figure 4. Two snapshots from the prototype implementation.
a) an in-world view, b) an out-world view.

Both collating abstractions into a single structure and slicing new abstractions requires a comparison and mapping of the respective languages and vocabularies and the (information) structures expressed in these. *Sorts*,

an approach to representational flexibility (Stouffs and Krishnamurti, 1997), provides support for this. Similar to XML, *sorts* specifies a common syntax, allowing for different vocabularies and languages to be created, compared, and related. Different sorts can be separately conceived for each abstraction and, subsequently, compared for similar components. Building on these similarities, and using an iterative process of development and comparison, a common sort can be arrived at that allows for all abstractions to be represented into a single integrated information structure. Once such a sort has been achieved, the individual abstractions can be mapped onto this sort and their results merged into a single composite structure. New abstractions can be extracted from this structure in a similar process, by conceiving an appropriate sort and mapping this onto the unified sort. No longer restricted by the original abstractions, querying an architectural design or analysis depends on an appropriate expression of the query as a sort, and interpreting the resulting information structure.

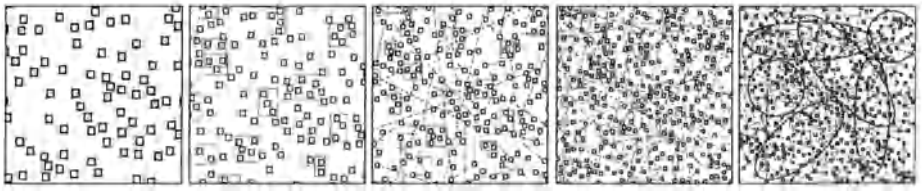


Figure 5. The integrated structure of a collection of abstractions. a) components, b) components grouped into meta-components, c) relationships between components, d) relationships between components and meta-components, e) abstractions distinguished within the network of components and relationships.

A sort is defined as a composition of other sorts under formal compositional operations, elementary data types define primitive sorts (Stouffs and Krishnamurti, 1997). Examples of such compositional operations are an operation of sum, allowing for disjunctively co-ordinate compositions of sorts, where each sort may – but does not have to – be represented in the data form, and an attribute relationship, providing for (recursively) subordinate compositions of sorts in both one-to-many and one-to-one instantiations. Other compositional operations are also considered, such as an array- or grid-like composition of sorts. The definition of a sort also includes a specification of the operational behavior of its members and collections thereof for common arithmetic operations. This behavioral specification enables a uniform handling of forms of different sorts. A primitive sort has its behavior assigned in order to achieve a desired effect; a composite sort receives its behavior from its component sorts, based on its compositional relationships (Stouffs and Krishnamurti, 1997).

Corresponding the representational format of an abstraction, a sort is defined from an appropriate selection of primitive sorts and a definition of its composition using formal operations. The formal character of these operations enables the recognition of formal relationships between different compositional structures that provide for the comparison and mapping of the respective sorts. Comparing sorts also informs on potential data loss when converting information structures between these sorts. The behavioral specification of sorts supports the mapping of data onto and between different sorts such that the resulting information structures are conform to the definition of the respective representations or sorts. Thus, different abstractions can be mapped onto a single sort and their information structures integrated without information loss, provided a careful selection of the sorts involved. Deriving a new abstractions corresponding a specific vocabulary follows the same process of development of a sort and of mapping the global information structure onto this sort.

5. CONCLUSION

Analysis plays an important role in design and education. An information structure that integrates the different aspects of the analysis, such that the analysis can be interpreted and used in ways other than the original aspects or abstractions present, would be particularly useful in an educational setting. Furthermore, as the World Wide Web gains more importance in all fields, including collaboration in educational projects, providing software that makes it possible for team members scattered over diverse sites to share and manage information while maintaining a comfortable, easy-to-use interface becomes crucial. It seems to us that enriching the information structure both by detailing the components and by tightening the structure through content relationships would provide a more powerful structure in such a system. Especially in analysis, one is not just interested in one or more specific documents to be processed or built upon, but in interpreting the structure seeking information related to a concept of interest. In such a context, a rich information structure where the views one can derive are not simply defined by the original documents is particularly worthwhile.

We believe that XML as a structuring language is specifically suited to define and develop this information structure when dealing with otherwise unstructured information, such as texts and images. However, XML does not provide any information on how to manipulate its data and, as such, is ill-suited to represent complex geometrical data. Analyses commonly include other data formats than texts and images, such as drawings, models, and numerical analyses. Integrating these into a rich representation suggests

a different representational language and requires a different approach for decomposing the abstractions into conceptual entities and for recognizing the relationships between these entities, both dependent on the format. XML and *sorts* both offer a hierarchical description of a data structure in terms of substructures, and enable the mapping of data structures by comparing their hierarchical descriptions. The conceptual framework behind *sorts* offers better support for complex entities, as it allows a behavioural specification of entities. This prompts us to develop the comparison and mapping of data structures using sorts, in order to take advantage of the additional capabilities of this framework. By comparing and evaluating the two presented approaches, resulting in two different applications, we also intend to extract key requirements concerning the representation and presentation of architectural analyses.

The final goal of the project is to derive at a specific implementation, yet from general principles (Tunçer and Stouffs, 1999). It is not the intention to develop a global system that can deal with all abstractions belonging to all sorts of building projects, but to define the representational framework for achieving an integrated structure of components and relationships from a collection of abstractions. The definition of abstractions and mechanisms can then be interpreted and implemented for different building projects or architectural bodies.

6. REFERENCES

Akin, O., M. Cumming, M. Shealey, and B. Tunçer, 1997, "An electronic design assistance tool for case-based representation of designs", *Automation in Construction,* 6, p. 265-274.

Egli, H.G., 1997, *Sinan: An interpretation*, Ege Yayinlari, Istanbul.

Erzen, J.N., 1996, *Mimar Sinan: Estetik bir analiz*, Şevki Vanli Mimarlik Vakfi Yayinlari, Ankara.

Papanikolaou, M. and B. Tunçer, 1999, "The Fake.Space experience - exploring new spaces", in: Brown, Knight and Berridge (eds.) *Architectural Computing: from Turing to 2000*, eCAADe and The University of Liverpool, Liverpool, UK, p. 395-402.

Schmitt, G., 1993, *Architectura et Machina: Computer Aided Architectural Design und Virtuelle Architektur*, Vieweg, Braunschweig, Germany.

Stierlin, H., 1998, *Turkey: From the Selçuks to the Ottomans*, Taschen, Cologne.

Stierlin, H., 1985, *Soliman et l'architecture ottomane*, Office du Livre, Fribourg, Switzerland.

Stouffs, R. and R. Krishnamurti, 1997, "Sorts: A concept for representational flexibility", in: Junge (ed.) *CAAD Futures 1997*, Kluwer Academic, Dordrecht, p. 553-564.

Tunçer, B. and R. Stouffs, 2000, "Modeling building project information", in: Gudnason (ed.) *Construction Information Technology 2000*, Icelandic Building Research Institute, Reykjavik, Iceland, p. 937-947.

Tunçer, B. and R. Stouffs, 1999, "Computational richness in the representation of architectural languages", in: Brown, Knight, and Berridge (eds.) *Architectural Computing: from Turing to 2000*, eCAADe and The University of Liverpool, Liverpool, UK, p. 603-610.

Constructive Representation in Situated Analogy in Design

An essay on a bottle of Eau d'Issey pour Homme

Jaroslaw M. Kulinski and John S. Gero
University of Sydney

Key words: Design, Analogy, Situatedness, Knowledge representation

Abstract: A model of situated analogy considered within the context of design is presented. It shows the impact of constructive knowledge representation on analogy making. The importance of a non-fixed but constructive representation is highlighted. It is suggested that a situated model of analogy fits the observed characteristics of design better than a non-situated one.

1. ARTEFACT ANALYSIS

This part presents the situated nature of design through an inductive analysis of a hypothetical design scenario used in the creation of an artefact. This design scenario is constructed using analogical reasoning. It does not aim to replicate the actual design process that was employed, rather it gives a plausible explanation of how the final result might have been reached and it serves as a scaffolding for a theoretical model that is the subject of the latter part of this paper.

1.1 A bottle

A recent example of industrial design: a bottle of perfume of Eau d'Issey is presented in Figure 1. The particular shape of this container reflects the specific aims it was developed for, i.e., a container for two complementary cosmetic products in the form of a travel kit. Let us try to analyse the design process that hypothetically could lead to this very specific form. Since the

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 507–520.
© 2001 *Kluwer Academic Publishers.*

subject of this research is focused on analogical reasoning, the scenario invented here is based on the application of analogy as a design strategy.

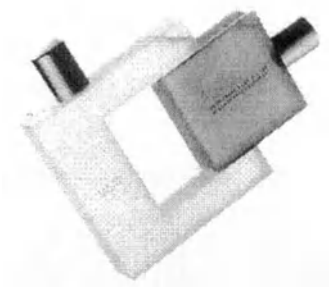


Figure 1. Karim Rashid: L'Eau d'Issey pour Homme; 2-in-1 travel kit; blow moulded polypropylene bottles with adflex coating

The design process as defined by Gero (1990) is a process of the production of a description of an artefact that meets certain functional requirements (the brief). Functional demands that are related only to the physical performance of an artefact form a core of the artefact specification. In this example, they can have a following form: provide a portable container(s) for two liquid substances. The abstracted form of a brief is useful as a search strategy for a prototypical solution to the design problem. Thus, a prototypical answer to the design problem at hand is a bottle.

However, placement of the design within the actual context reveals a much richer set of functional demands. In the example the product that is to be carried by the container being the subject of the design process is a perfume, moreover it is targeted at male users. Thus, once placed within any cultural context it reveals a much richer cultural functional set. In this design the bottle is not only a container but also it is a signifier of the other contextual meanings. The bottle of perfume is then not a simple container it is also a cultural marker.

The reason for this digression is to make the reader realise how easily a human subject is able to re-classify an artefact, or using more computational terminology how easily one can re-represent an object when considered within a specific context.

The following analysis makes an extensive use of re-representation as a method that contributes to the development of the design.

1.2 The "zipper" bottle

The little bottle that is considered here was conceived as a travel kit. In the context of travel the functional requirements are related to efficient space utilisation, the reduction of the number of items to take care of, and shock resistance, travel also introduces the notion of movement and speed. These aspects can be recognised in the design form.

The design of the bottle is the physical expression of compacting, i.e., a single artefact yet serving a double purpose. The 2-in-1 bottle design reinvents the bottle not as a container but as a package that conveniently reduces the number of individual items, i.e., less things to forget. Let's trace hypothetically how these elements could have been incorporated within the design of this double bottle. For an artefact, the feature of being compact hinges on the ratio between the actual form and its bounding box, i.e., the more the actual shape resembles its bounding box the ratio of its compactability is higher. The subject of this example apart from being individually compact must also be made out of two parts (2-in-1). The issue was to create a form that is singular yet dividable into parts. And further, the least tangible aspect of the representation, it has to embody the notion of travel.

To recapitulate, the few behaviours that we are seeking in this design are: 1. a single artefact though dividable, 2. a shape that easily fits other objects, 3. bringing in-line the broad cultural notion of travelling.

There are number of methods that can be applied to achieve the requirements, but expressively we are using only one, that is, analogy. Let us imagine than we scan through a set of other artefacts for an object having a matching set of behaviours. One of our candidates could be a zip fastener. A zip fastener, functionally, is a device that connects two in one, i.e., without modifying its own structure it snuggles its part together in one. It is made as two collections of teeth that fit into each other. Also for a human designer, a zip fastener fits within the broad category of an object relating to travelling (ex. travel baggage).

The objective of analogy in design is to extend the design state space by transferring new behaviours (and associated structure features) from the source design to the target. In this example the target design is the double bottle, the zipper is its matched source. One of the behaviours that is unique to the zipper is the snapping together of its parts. This behaviour can be transferred to the double bottle design. This requires structure modification of the singular parts into tooth shaped elements that can snap together.

This example aims to present the issue of representation as a factor that allows a novel analogical match between two objects. In the subsequent paragraphs this intuitive example will be investigated more rigorously giving

a general model of situated analogy in designing. This model is defined as situated since at the core of its operation lays the feature of constructive representation.

2. DESIGN, ANALOGY AND REPRESENTATION – THEORETICAL FRAMEWORKS

Analogy is used as one of the strategies employed in the design process that has been recognised as one that can change the design state space. (Gero 1990, Gero & Kazakov 1999, Navinchandra 1991).

2.1 FBS model of design

According to the model of design proposed by Gero (1990), the design process is a task of producing a description, D, of the structure, S, which responds to functional requirements, F, through expected behaviours, B. In order to accomplish these design tasks one has to define the mapping between these three states. In such a framework it can be said that a design is represented as a triplet composed of those three states: function, behaviour and structure (FBS). The FBS triplet can also be defined as causal and abductive knowledge, since it represents a mapping between the design structure onto a set of behaviours that such a structure presents and onto a set of functions that can be attributed from the behaviours, and vice-versa. The use of this model provides an opportunity to represent any design as a graph of relations between states. Throughout the rest of this paper making a "design representation" means creating a graph of transitions between these three states.

2.2 SME framework for analogy

The model of analogy adopted in this research was proposed by Gentner (1983) (also Falkenheimer, Forbus & Gentner 1989) and termed Structure Mapping Theory. The important feature that makes this approach usable within a computational environment is the focus on syntactic properties of knowledge representation, and not on a specific content of the domains. The central proposition of the structure mapping engine (SME) is that analogy is a mapping of knowledge from one domain (the base or the source) into another (the target) which conveys that a system of relations known to hold in the base also holds in the target. The target objects do not have to resemble their corresponding base objects. Objects are placed in correspondence by virtue of corresponding roles in the common relational

structure. The model that we propose is an extension of SME that includes the issue of how the actual representation for each of the analogical domains is constructed.

2.3 Situated perspective on analogy

Analogy-making relies on matching two domains that share common representational structures. This requires us to consider knowledge representation. If the knowledge that is stored in long-term memory follows the encyclopaedic paradigm, that is, the knowledge is represented in as fixed, self-contained chunks that can be uniquely addressed and retrieved, analogy making is reduced to a search and retrieve process. Such a conclusion, however, stands in clear opposition to everyday experience in which analogy is acknowledged as the ubiquitous mechanism by which humans understand abstract concepts and carry out abstract reasoning (Lakoff & Johnson, 1980 in Nehaniv, 1999).

In order to escape this search-retrieve model, it has been postulated that analogy-making relies on seeing two domains as "the same" (Chalmers, French & Hofstadter, 1995), that is, finding means of representation that allow matching two domains with respect to specific criteria. In this view making analogy hinges on the ability to re-represent the knowledge in a flexible, i.e., adaptable to needs, manner.

The situated cognition perspective postulates an alternative to the encyclopaedic paradigm of knowledge representation. From the situated standpoint, the setting (i.e. the actual context and its perception) within which human actions develop, shapes the representation of knowledge to its own needs. It is claimed that what people perceive, how they conceive of their activity and what they physically do develop together (Clancey, 1997). As a result of this standpoint a different paradigm of human memory is proposed: it is no longer a repository of ready made structures but rather the memory provides the capability to produce structures, called representations. (Clancey, 1991; Rosenfield, 1988). By applying the notion of constructive representation within analogy-making we are able to meet the postulate of tailor-made domain representations.

2.4 Design as a situated activity

A design process that is not limited to optimisation only, is seeking novel means of meeting the functional requirements of the brief. Schön & Wiggins (1992) observing his own and other designers' actions concludes: that during the design process under the pressure of perceptual processes an alternative view is developed on the problem at hand. He calls this

phenomenon as "reflection in action". The acknowledgement of this novel view allows inclusion of the emergent features that were not accounted prior to the design process starting.

Partial support for this view can be found in the protocol studies of designers. Suwa, Gero & Purcell (1999) have found that the design requirements that drive the process are built inside the design process itself. They examined the notion of unexpected discoveries (UXDs) that occur to the designer during the design process. They looked at their correlation with the formulation of more general design strategies called situated-inventions (S-inventions). S-inventions are defined as the generation of inaugural issues or requirements in the current design task. Since such formulations occur during the design process they are situated in this process. The S-inventions form the basis for the formulation of new design goals. The study pointed out an important bi-directional causality between unexpected discoveries and the formulation of design goals. A conclusion of this study suggests that the pursuit of various design goals is mutually interwoven. A solution found for a particular set of design goals may potentially trigger reformulation of the design problem through the generation of alternative goals.

Figures 2 and 3 show the reflective process and its consequences on the development of the design process. In Figure 2 the design process is considered as a linear solution process that progresses from the initial problem to the conclusion in one pass.

Figure 2. Initial stage of the situated design process: problem framing

Figure 3 presents the subsequent stages of the process, i.e., the reflective component. The reflective process starts where the preceding phase ends, i.e., with the representation of the artefact (rounded corner box in Figure 3 indicates the new starting position). The comparison of the perception of the design representation in the brief and in the artefact is responsible for finding the emergent features (UXD) in the latter one. These differences lead to modified design goals and subsequent re-representations of the design artefact. The dotted arrow from the last representation of the artefact indicates that it is a cyclic process that can be repeated an arbitrary number of iterations.

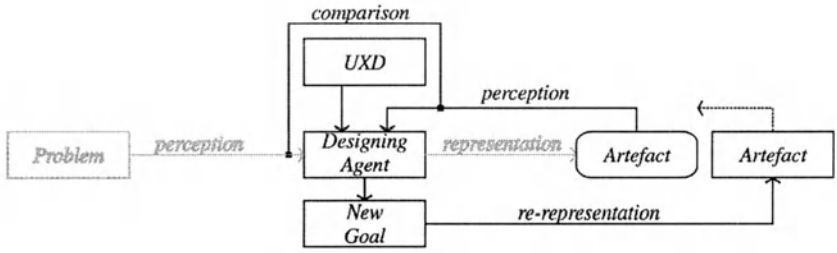


Figure 3. Reflective stage of design process
Greyed elements indicate those in the preceding phase.

3. A MODEL OF SITUATED ANALOGY

There are two aspects of situatedness in this model; the first is related to the mode of the construction of the representation of analogical domains and the second is related to the whole design process as described earlier. Figures 4 to 7 present in a time sequence the analogical process within the framework of situated cognition. Elements greyed out indicate that these particular ones do not take part in the episode. Various types of lines indicate logically continuous flows within a single episode.

Figure 4 presents the initial problem framing, in the case of analogy, it is finding and representing the target design.

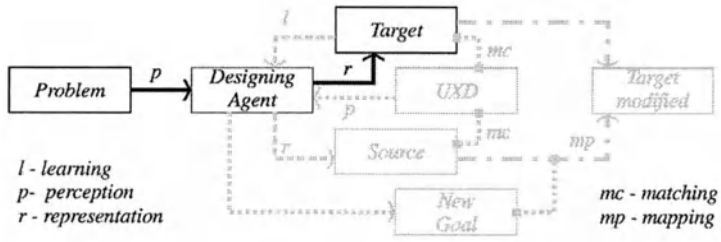


Figure 4. Target representation

Figure 5 presents the continuation of the analogical reasoning, i.e., a search for a source candidate that can match the target representation. For the sake of clarity this diagram is substantially simplified and presents only one potential source candidate. The search for an appropriate source candidate starts from the target representation. The learning link from the target indicates that the process of constructing a target representation has been learnt by the agent.

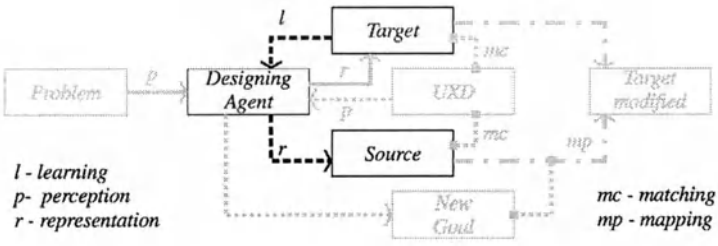


Figure 5. Source representation

This knowledge in turn is applied when the designing agent attempts to construct a matching source representation. In this learning process the form of the source representation is dependent on the target representation: without the target representation there would be no means of constructing the source representation.

Figure 6 shows the consequences of finding an analogical match between the target design and source design. Referring back to the earlier example, the finding that two bottles in a travel kit can be matched against a zip fastener suggests a discovery.

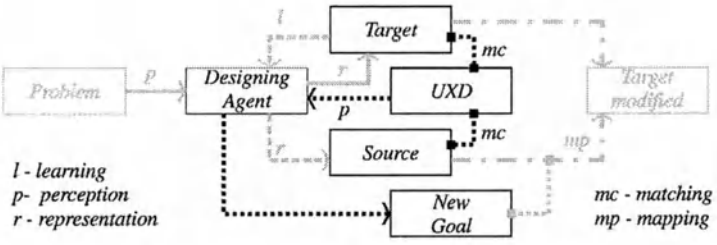


Figure 6. Analogical match

Figure 7 presents the final stage of a single reflective episode in a design process, i.e., the mapping of the analogical match. Mapping in terms of analogy means the process of transferring certain structural features found in the source but not present in the target, that is, modifying the target structure. As in the example a modification of the bottle shape that would provide that "snapping" behaviour.

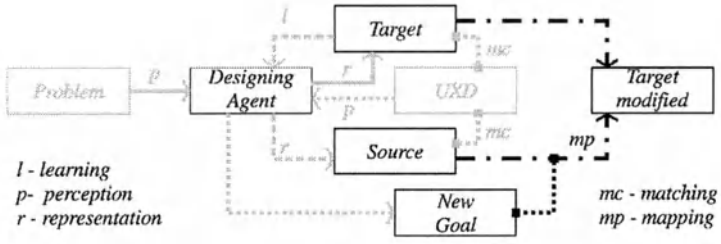


Figure 7. Analogy mapping

4. SYSTEM ARCHITECTURE

The computational system is not developed as an autonomous design system but rather a mind aid for a human designer. Thus, at different stages of operation human input is expected in order to drive the process. The operation of the system can be described as an interaction of 7 different agents. Figure 8 shows the flow of data between the agents. The process is initiated by the Target Chooser that on the basis of a brief analysis selects the target design. It means that from the knowledge base of known designs a target structure is selected. The next step in the process is the construction of the FBS graph of the target design. Through analysis of the brief a context of application is selected.

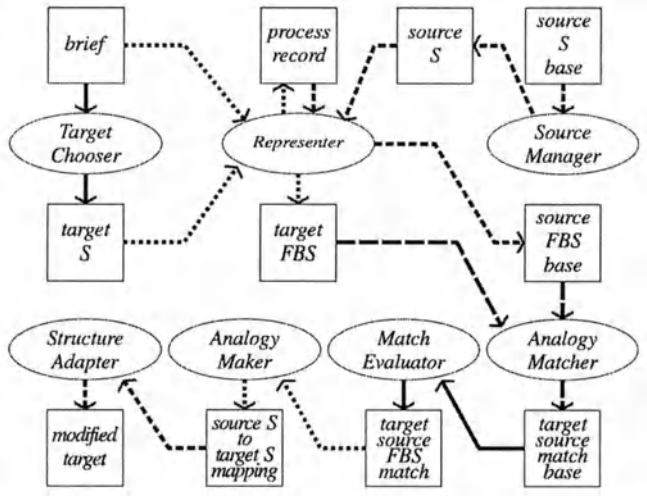


Figure 8. System architecture

This context determines the scope of functions that the target structure is to meet. The function set allows the derivation of the set of expected

behaviours that implements the functions. The Representer uses the context derived expected behaviour set and inspects the target structure for features that support the expected behaviour. The set of successful behaviours found in the target structure is recorded. Subsequently it is used as set of expected behaviours when the source candidates are represented. The role of the Source Manager is to run the representational process for each of the source structures in the knowledge base of familiar designs. The product from this process is a base of potential source candidates. The role of the Analogy Matcher is to find the most successful matches between target and source candidates. For a single pass of the system the Match Evaluator selects only one matching pair for further processing. On the basis of this selection the Analogy Mapper proposes an analogical conjecture between the target and the source structures. This conjecture forms the basis for the Structure Adapter to modify the target structure in order to transfer additional features found in the source structure to target structure.

5. IMPLEMENTATION

Here we present only the implementation of the Representer. The construction of the representation process is built around a queue that uses and manages small software agents that can detect design structure features that can support a particular behaviour. Each system behaviour has an associated perceptual agent.

5.1 Elements of the Representer

5.1.1 Knowledge base of familiar design

In the knowledge base the designs are represented only partially, that is, only their structure is encoded. The structure representation remains static throughout the whole process. The structures are composed as a hierarchy of compounds, parts, relations and attributes. On a compound level each design is encoded with all its constitutive parts and topological relations between them. In turn, on a part level each item is considered as an independent entity and the description stored encodes only its physical properties. For example a partial encoding of a bottle design is represented as follows:
on a compound level:

parts(bottle): {bottom, side, neck, cap}
orientation(side): {vertical}
relation(side, bottom): {perpendicular, fixed}
relation(side, neck): {fixed}

relation(side, neck): {bigger}
on a part level:
shape(side): {void, rectangular, continuous, axial}
material(side): {plastic}
...

The knowledge base of design structures is the area in which behaviour detecting agents operate.

5.1.2 Base of known behaviours, functions & contexts

This base is arranged as a graph of relations where there are three categories of nodes, i.e., behaviours, functions and contexts (BFC). This network can be read as mapping between these elements. An arc that links a function *n* and a context *m* can be read as function F_n *is relevant in* context C_m. The arcs between behaviours and functions can be read as: function F_n *is implemented by* behaviour B_m.

5.1.3 Behaviour detecting agents

All behaviours present in the BFC network are associated with respective software agents. These agents are equipped with the rules that can examine design structures encoded in the knowledge base of familiar designs. These rules attempt to determine whether a particular structure presents a particular behaviour or not.

Apart from the detection rules, the agents are also equipped with rules that mediate the relationship between the agents themselves. Thus, there are effects of agent actions: 1. a success or failure in the behaviour detection process and 2. an input into the selection process of agents activation. The latter action is dependent on the result of the former.

5.1.4 The queue processor

The queue processor is the manager that orders and triggers the action of behaviour detecting agents. An overview on the main processing queue is shown in Figure 9.

When a set of behaviour detecting agents is posted to the queue it is ordered and executed in a sequential manner. Due to the nature of ordering in the queue some of the agents may not be able to pass the activation threshold that is required to become executable. These agents are removed from the queue without being processed and they fall into the "discarded bin". The ones that pass the activation threshold after being executed fall into either successful or unsuccessful bins. They are able to post to other

related agents to the queue, shown in Figure 9 as the *in-process batch*. The process terminates when all posted agents in the queue are exhausted.

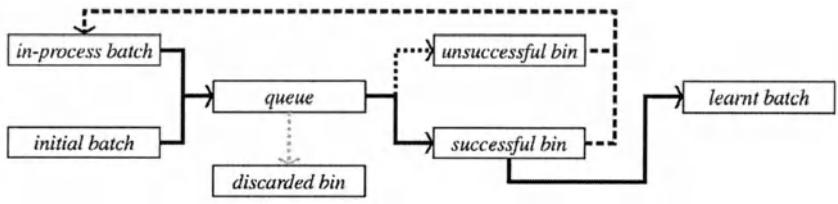


Figure 9. Process queue in representer

The order of execution of the agents in the queue is dependent on their level of activation. The activation of an agent is calculated on the basis of two parameters: the initial confidence factor and the age of posting. These two factors work in an antagonist manner, that is, the age lowers the confidence factor. The age factor constantly increases during the queue processing. It is measured in process cycles of equally valued items in the queue. As a result the agents that were posted with low confidence level have very little chance to become executable, since the in-process posted agents are always younger. The rationale for this process is grounded in an assumption that in a situationally driven process the objective is to search only for the most situation-relevant results and not to proceed with a wide search for all possibilities. The in-process posted agents are the followers of the initially selected ones. This phenomenon of following can be interpreted as a self directing search since the development of the process serves as a reinforcement factor for the direction of the search.

5.2 Operation

The representation process is initiated by a selection of agents posted in the queue. The selection process differs in the case of target and source representation. In the case of the former the initial batch is context derived, while in the latter case it is the learnt batch that has been acquired in the course of constructing the target representation.

In the example presented earlier one of the expected behaviours of a bottle in the context of travel was efficient space occupation. The detecting rule for an agent seeking the efficient space usage has the following form: "If compound x (or sub compound x) has shape(x):{axial, rectangular} and orientation(x):{vertical or horizontal} and relation(x,*others*):{bigger} then x can have an efficient bounding box ratio". Success in meeting the above rule moves this agent to the "successful bin" and it will be re-applied when the source representation is constructed.

The positive finding by this agent also triggers the posting of a related agent such as the one that detects whether two compounds (sub compounds) can fit side by side. A rule for such an agent can be written as follows: "If x_1 and x_2 have shape(x_n):{axial, rectangular} and relation(x_1,x_2):{parallel} then x_1 and x_2 can fit each other." In turn this agent is related to the one that detects the fitting of top-bottom and so on.

The queue manager prevents the circularity of postings, that is, an agent can only be accepted once.

6. CONCLUSIONS

The focus of this research is the conceptual stage of the design process. Design studies show that on this level the design process is not completely defined in terms of constraints that shape the final product, rather they are built along with the ongoing progress of designing. These claims inspired the hypothesis of the importance of the constructive approach to knowledge representation. A system that employs this paradigm in knowledge representation is presented and examined within a context of analogical reasoning.

6.1 Validation

In order to validate the hypothesis the same analogy engine is intended to run in two distinct modes: one when it is connected to the fixed represented database of familiar designs, two when it is inserted into the system as described earlier. It is expected that the runs of the engine in each mode should produce detectably different results. Especially, it is expected that the run that employs constructive representation should generate richer outcomes both quantitatively (more conjectures) and qualitatively (more novel or more surprising conjectures).

6.2 Application

The system presented here is considered to be a mind aiding tool. Its sole purpose is to provide the designer with a potentially novel analogical conjecture. Thus, this system can be seen as a sort of "brain storming" tool that the human designer can employ as an assistant. This system, although without the ability to understand the complexity and semantics of the design process, attempts to provide a framework for a tool that can behave as "a stupid savant" which requires human intervention and understanding to properly interpret and use the proposed conclusions. The system cannot

solve design problems on its own, yet one can expect that by engaging the designer in a discourse throughout the whole design process, the resulting dialog should facilitate finding novel design solutions.

7. ACKNOWLEDGEMENTS

This work has been supported by an Overseas Postgraduate Research Award and by an Australian Research Council grant.

8. REFERENCES

Chalmers, D., R. French and D. Hofstadter, 1995, "High-level perception, representation and analogy: A critique of artificial-intelligence methodology", in D. Hofstadter , *Fluid Concepts and Creative Analogies*, Basic Books, New York, chapter 4.

Clancey, W. J.,1991, "Book review: Israel Rosenfield: The Invention of memory: A New View of the Brain", *Artificial Intelligence* 50, p. 241–284.

Clancey, W. J., 1997, *Situated Cognition: On Human Knowledge and Computer Representations*, Cambridge University Press, Cambridge.

Falkenhainer, B., K. D. Forbus and D. Gentner, 1989, "The structure-mapping engine: Algorithm and examples", *Artificial Intelligence* 41, p. 1–63.

Gentner, D., 1983, "Structure-mapping: A theoretical framework for analogy", *Cognitive Science* 7, p. 155–170.

Gero, J. S., 1990, "Design prototypes: A knowledge representation schema for design", *AI Magazine*, 11(4), p. 26–36.

Gero, J. S., and V. Kazakov, 1999, "Using analogy to extend the behaviour state space in creative design", in J. S. Gero & M. L. Maher (eds), *Computational Models of Creative Design IV*, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, Australia, p. 113 – 143.

Lakoff, G., and M. Johnson, 1980, *Metaphors We Live By*, University of Chicago Press, Chicago.

Navinchandra, D., 1991, *Exploration and Innovation in Design*, Springer-Verlag, New York.

Nehaniv, C. L. 1999. "Computation for methaphors, analogy, and agents", in C. L. Nehaniv (ed.), *Computation for Methaphors, Analogy, and Agents*, number 1562 in Lecture Notes in Artificial Intelligence, Springer, Berlin; New York, p. 1–10.

Schön, D. A. & G. Wiggins, 1992. "Kinds of seeing and their functions in designing", *Design Studies* 13(2): p.135 – 156.

Suwa, M., J. Gero and T. Purcell, 1999, "Unexpected discoveries and s-invention of design requirements: a key to creativity in designs", in J. S. Gero & M. L. Maher (eds), *Computational Models of Creative Design IV*, Key Centre of Design Computing and Cognition, University of Sydney, Sydney, Australia, p. 297–320

Rosenfield, I., 1988, *The Invention of Memory: A New View of the Brain*, Basic Books, Inc. Publishers, New York.

Cognition-based CAAD

How CAAD systems can support conceptual design

Hsien-Hui Tang and John S Gero
The University of Sydney

Key words: design cognition, protocol analysis, conceptual design, CAAD

Abstract: This paper introduces the concept of cognition-based CAAD. Protocol analysis and a content-oriented coding scheme are utilised to produce cognitive results of designers' behaviour. This empirical analysis suggests that the speed of thought and vagueness among actions are the main areas to be supported by any cognition-based CAAD system. Three different modes of design thinking are presented as the basis of a possible CAAD system.

1. INTRODUCTION

Current commercial CAAD systems focus on representations and manipulations of those representations with a small number being able to deal with information processing views of designing (Gero, 2000). This paper proposes that, from a cognitive point of view, perception, functional references, and inter-linked relationship between high and low cognitive levels should be integrated into CAAD systems to produce a cognition-based CAAD system. We utilise empirical results obtained by examining designing behaviours of expert and novice architectural designers to substantiate our assumptions. The results imply that these areas should play important roles in future CAAD systems in order to allow them to be used during conceptual design. The empirical results are based on protocol studies of an expert and a novice designer. The remainder of this paper describes the experiments and the results and places them in the context of a potential CAAD system.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 521–531.
© 2001 *Kluwer Academic Publishers.*

2. METHODS

Protocol analysis is utilised to examine the design processes in order to provide information for a cognition-based CAAD system. Recently, retrospective protocols has been used in several studies to explore human design activities because of the claim that they minimise the interference they cause to the designers. This study follows the same method in which participating designers first design for the designated brief, and then retrospectively report the design process with the aid of the videotapes of their designing in the first phase.

The oral report is transcribed into a raw protocol. Utterances, sketches, and videos are the material by which researchers examine and understand the design behaviours and encode the protocol. The encoded protocol is organised into information processing streams, documenting which types of information and how they have been processed in terms of our coding schemes.

The analytical structure of this coding scheme consists of two primary levels. Lower level cognitive activities contain physical actions and perceptual actions that interact with the outside world, including drawing, looking, and recognising graphical features and spatial relationships. Higher level cognitive activities contain functional references and conceptual actions that interact with the designer's internal world, including functions, views, setting up goals and making decisions.

This coding scheme was originally established by Suwa and Tversky (Suwa and Tversky, 1997; Suwa, Gero, et al., 2000), and applied by Tang and Gero for their series of studies (Tang and Gero, 2000). It represents the design process in terms of four inter-linked and inter-related types of cognitive actions – physical, perceptual, functional, and conceptual actions in each segment. An episode of the design process is constructed by a series of consecutive segments. These segments represent the smallest units of intentions of a designer. All the four types of cognitive actions in one single segment have relations but no chronological order.

3. CONCEPTUAL DESIGNING

Conceptual designing occurs when the designer is trying to understand the problem and set the situation for the following processes. It does not mean that the designer does not exhibit similar features during the remainder of the design process, but conceptual designing is the period having the richest range of ideas, problems, and creativity. However, current CAAD systems seem to be absent from this design process, and most designers still

conduct their conceptual designing using hand-drawn sketches, or more romantically on the back of an envelope.

Given that CAAD systems should have the capacity to support this design phase to provide a holistic design process, from concept generation to manufacture, this study examines what CAAD systems should support in order to support designers during conceptual designing.

3.1 Speed of shifting intentions

We assume that the protocol is divided according to the designers' intentions, instead of verbalisation events or syntactic marks (Ericsson and Simon, 1993). This has been applied in recent protocol studies in which designers' intentions are understood not only through the verbal utterances but also through their drawings and gestures (McNeill, Gero, et al., 1998)). In the same sense, one segment consists of pieces of information, which appear to have occurred simultaneously in the designer's mind, and constitute a set of coherent cognitive actions – physical, perceptual, functional, and conceptual. Thus, segments are the representation for the designer's intentions, and the speed of shifting intentions represents the speed of thought and related actions.

In our empirical data, shown in *Table 1*, the expert's encoded protocol consists of 338 segments with an average time span of each segment of 8 seconds, whereas the novice's encoded protocol consists of 145 segments with an average time span of 20 seconds. The experimental duration was 45 and 48 minutes respectively.

Table 1. Number of segments and average time span of the novice and the expert

	Number of segments	Average time span (seconds)
Novice	145	20
Expert	338	8

The result indicates that this expert shifted his focus from one topic to another on average every 8 seconds, and the novice did so every 20 seconds. Given that the speed of shifts represents that of thought, it is much faster than was expected. In terms of using a CAAD system, the time this expert took to shift his focus was of the order of that for a user to pull down a menu, select the function, and input parameters.

The surprisingly fast speed of change of topic during conceptual designing provides a basis for why designers still prefer using pen and paper even when they have expensive, powerful, and cutting-edge CAAD systems, during conceptual designing. It is simply because these familiar sketching skills can catch up with the speed of thought, the speed of ideas, and the speed of creativity. The speed here is not relevant to computational power

nowadays because even the latest CAAD system cannot efficiently support this design phase. The problems are that the interface between designers and machines is not sufficiently intuitive and simple enough to follow the train of thought so that the use of a CAAD system blocks the development of thought and ideas. It results in spending too much time on the software rather than on designing.

3.2 Speeds of physical, perceptual, functional, and conceptual actions

In our encoded data, every segment consists of four different kinds of actions, but the number of actions in a segment depends on the content of that segment. For example, one segment may have 4 physical actions, 2 perceptual, 3 functional, and 1 conceptual, while the following one may have only 2 physical and 1 perceptual actions.

"Physical" refers to actions that have direct relevance to depictions via drawing, revising or gesture. "Perceptual" refers to actions that attend visuo-spatial features or relations of depictions with or without physical actions. "Functional" refers to the actions that attach functions and abstract concepts in depictions or visuo-spatial features and relations. "Conceptual" refers to the actions that manipulate functions and abstract concepts through setting-up goals, aesthetic evaluation, and design knowledge. These four actions and their relations capture the essence of the cognitive processes of design thinking.

Table 2 and *Table 3* show the basic statistical descriptions of each type of actions in our novice and expert's encoded data. The results indicate that on average the novice has 3.6 physical and 1.5 perceptual actions every segment, which spans 20 second on average. Whereas the expert has 3.1 physical and 1.8 perceptual actions every segment, which lasts 8 seconds on average. These external action rates were measured when designers were using pen and paper.

Table 2. Basic Statistical Description of actions in the novice's encoded protocol

	Mean	Std. Dev.	Minimum	Maximum
Physical action	3.6	2.3	0	15
Perceptual action	1.5	1.2	0	5
Functional action	1.9	1.6	0	7
Conceptual action	0.8	0.5	0	2

Table 3. Basic Statistical Description of actions in the expert's encoded protocol

	Mean	Std. Dev.	Minimum	Maximum
Physical action	3.1	1.5	0	8
Perceptual action	1.8	1.1	0	6
Functional action	2.3	1.6	0	7

	Mean	Std. Dev.	Minimum	Maximum
Conceptual action	0.8	0.8	0	4

The average numbers of all actions in every segment further emphasises the rich information processing that occurs during conceptual designing. According to this result, it is hard to image how designers can use current CAAD systems during conceptual designing. For example, in 20 seconds how may operational actions can a novice finish in a CAAD system whether using pull-down menus or short-cut keys or not. In terms of cognitive load of human memory (based on the 'magic number seven' for short term memory capacity), there is very little capacity left for designers to use the tools if we intend to maintain the same speed. As a result, a cognition-based CAAD system would be one with a very simple and intuitive interface without many selections and pull-down menus.

Moreover, examining current CAAD systems from the perspective of these four actions, we can find that most CAAD systems focus on two actions: drawing in physical actions and design knowledge in conceptual actions. The former are representational kinds of CAAD systems, such as AutoCAD and ArchiCAD, with their emphasis on the scaled details and presenting all the components in a two- or three- dimensional world. The latter ones are knowledge-based or case-based CAAD systems, providing designers possible and useful design knowledge. Very few systems are capable of supporting a designer's perceptual and functional actions. However, they are as important as drawing and design knowledge in terms of frequency of appearance during conceptual designing.

3.3 Vagueness

Vagueness and uncertainty of depictions and their relations with other types of design activities during conceptual designing are the other feature resulting in the failure of current CAAD systems to adequately support this preliminary design process. Goel (1999) regarded this feature as the characteristics of ill-defined representation, being imprecise, ambiguous, fluid, indeterminate, etc.

Drawing in physical actions are the foundational and most frequent activities according to our analytical results. All the actions are relevant to or based on drawing and sketches; Schön and Wiggins have long described the design process as the reflective and conversational process between designers and sketches (Schön and Wiggins, 1992).

Most of the sketches during conceptual designing are like *Figure 1* in terms of vagueness. In these sketches, depictions are syntactically uncertain, ambiguous, overlapping one with another, and scale-wise unfixed.

Numerous shapes and spatial relations could be generated from an arbitrary combinations of these depictions. These qualities help designers create lateral transformations of ideas for creative problem solving in Goel's terminology. In terms of CAAD systems, the syntactic disjointness and differentiation of drawing systems, however, block the possibility for designers to emerge more ideas from revising their sketches. The sketches of our other subjects in this experiment indicate that CAAD systems for novices should be easier and their representations should be less fixed since they tend to test and refine their ideas more.

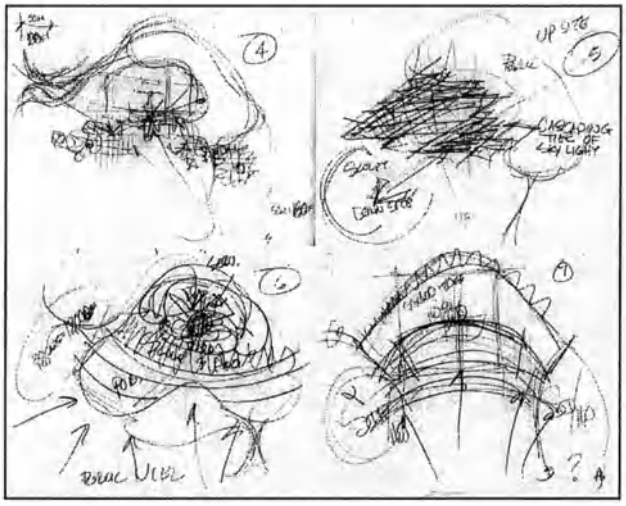


Figure 1. Numbers 4, 5, 6, and 7 of the expert's sketches

3.4 Multiple interpretations and relations between physical, perceptual, functional, and conceptual actions

This study confirmed that drawings or sketches have both perceptual and functional ambiguities and this may be why sketches play such important roles in the concept design process. In terms of perception and revision, the ambiguities provide opportunities to revise existing depictions and generate new spatial relationship from combinations as the source for aesthetic evaluations or form manipulation. Our data shows that during revisions designers regarded different depictions as one single element for functional manipulations although they were created individually at different times. Consequently, some depictions play several roles in different graphical and functional groups.

In terms of functional references, designers attached different functions to depictions without unchangeable fixations. They could be attached to overlapped areas and leave all the details for later consideration, or the functional references attached to depictions could evolve and change from time to time. There was no permanently fixed relation between depictions and functions in the concept design process.

This kind of vagueness and uncertainty really separates the conceptual design process from the rest of the design process, such as refinement and detail designs. During conceptual designing, designers play the essential role of confronting problems, generating ideas, and creating potential solutions, while during the rest of designing manufacturing and construction methods, building systems, and detailing occupy essential roles.

Corresponding to Norman's idea (Norman, 1998), the former part could be regarded as analogous to human-centre designing, and the latter part, to digital machine-centre designing. It is beneficial to apply CAAD systems to aid designers in generating the refinements and details of the design since it demands precision and accuracy for the mechanical process. CAAD systems successfully supported designers to realise structurally complex architecture that could not be built without the support of CAAD systems, such as the amazing works of Frank Gehry (Co and Forster, 1998).

What designers need to support conceptual designing are vague, imprecise systems to help them generate new ideas from their sketches. As a result, cognition-based CAAD means that at the early stage of designing such systems should support imprecision and indeterminism, but during the later stage it should be precise. Imprecision and indeterminism can be provided through a fluid representation schema that supports multiple views and multiple representations.

4. THREE MODES OF THE DESIGN PROCESS

After examining the design process from the viewpoints of speed and vagueness, we further examine the inter-relationships between physical, perceptual, functional, and conceptual actions. Based on the experimental data, we found that the design process may be viewed as a collection of modes, showing the directions of thinking process among four cognitive activities.

A sensor-driven process is one that is triggered by the designers' drawing and seeing. The reflective communication with the design media leads this process. A concept-driven process is one that is executed based on the designer's sub-goals or design strategies. The design media is used to realize ideas. A hybrid process is one in which perceptions and concepts interact

with each other, and the design media is used both to present results and to stimulate thinking.

4.1 Sensor-driven mode

Given the four cognitive actions, sensor-driven modes are triggered by physical actions, including drawing and looking actions, toward the conceptual actions via perceptual and functional actions. In this mode, designers search for ideas or solutions through drawings and revising existing sketches. Drawings here are doodles and arbitrary lines because of no specific thought leading them. This mode benefits from the uncertainty and vagueness of the depictions, which produce more spaces to imagine and generate new spatial relations for functional references and reasoning. In our encoded data, 42 percent of the expert's segments and 29 percent of the novice's were pure sensor-driven, meaning no conceptual action was involved in that segment.

CAAD systems intending to support this mode should be able to support the vagueness of these drawings, the essential quality of this process. Designers can produce any lines and curves without interference from the interface. Such CAAD systems should provide the basic characteristics of using pens and papers, such as speed, extremely simple interface, but the sketches should be stored and be able to be manipulated later.

4.2 Concept-driven mode

In contrast, concept-driven modes are driven by conceptual actions toward the physical actions via perceptual and functional actions. In this mode, designers have more specific intentions or decisions, and they use sketches to externalise their thoughts, integrate them into the existing depicted design, and examine the feasibility through perceptions. Drawings here are more certain and clear, and most of the time they are modifications or additions of the existing depictions. However, our empirical data found no pure concept-driven process, meaning no physical actions involved.

CAAD systems intending to support this mode will have to have a simple verbal entry for graphical information. This is because in our analysis designers tend to handle simple issues in segments as well as through a consecutive series of segments and then solve larger issues. The process is similar to Simon's well-known proposal that designers break a complex goal into sub-goals and then integrate the sub-solutions to solve the original problem.

4.3 Hybrid mode

The discussion of sensor-driven and concept-driven processes in this study is supported by the conceptual model proposed by Galle (1999). Our results fit into the structure of designer's artefact-idea and design representation in which two actions are named: production and interpretation. Our findings verify its basic structure, but also propose a question for the structure. Our experimental data shows the actions are not always clearly distinguishable. The characteristics of the design process are sometimes like a mixture of both actions, a hybrid mode. The mode is driven by physical or conceptual actions toward physical or conceptual actions via perceptual and functional actions. The essential feature is the feedback from themselves, and drawings here have both features of those in the previous two modes.

The main characteristic found in all these modes is that these four cognitive actions are closely linked in the design process. There are very few instances that physical or conceptual actions exist alone in a segment. As a result, the vague drawings system and the simple-entry graphical information support systems should be integrated together, being coherent and compliant with different modes. Moreover, sketch-related physical actions appear in every segment, so this process should be regarded as a visual reasoning process, rather than pure information processing.

5. DISCUSSION AND CONCLUSIONS

In this study, we first explored the speed and vagueness of the conceptual designing process through our empirical data and analytical coding schemes. These two characteristics are probably the reasons why current CAAD systems failed to support conceptual designing and the possible impetus for a future CAAD system. Such a future CAAD system should be cognition-based, i.e. based on the cognitive behaviour of designers, the subjects CAAD systems intend to aid.

We discussed the design process in terms of four cognitive actions. Current CAAD systems benefit presentational drawing actions in a very detailed way, but offer no support for perceptual recognition and functional attachments. The connections between the sketches and these higher level cognitive functions are absent. Our empirical data, however, shows they are as important as physical actions and the four cognition actions are not separable in the design process. In the last ten years, little has been done in this aspect since Ullman, Wood, and Craig (1990) suggested that CAD was

really only computer-aided "drafting", and was not capable of supporting sketching in any meaningful way.

Finally, we discuss the design process and CAAD from three different design modes. From the viewpoint of sensor-driven processes, a CAAD system should provide quick access to represent ideas but the representation should not be fixed, to allow for the emergence of a variety of unexpected forms and spatial relationships. In terms of concept-driven processes, a knowledge base should support the design to accompany the sketches or representations since no pure information processing was found in the design process. Most importantly, in terms of hybrid processes, future CAAD systems should be integrated systems, assisting both higher and lower cognitive activities.

In the design community, Gross's Electronic Cocktail Napkin project is a prototype diagramming environment aimed at conceptual designing process based on the similar ideas we proposed in this paper (Gross, 1996). In his CAAD environment, a pen-based interface supports the ambiguity and non-commitment required during conceptual designing, parsing and recognition systems analyse the sketches and their spatial relationships, constraint management routines keep the high-level relationship between diagrams. The purpose of that project is to support designers in the incremental formalization of the design process, from conceptual designing to schematic designing.

It is proposed, supported by our empirical data, that a cognition-based CAAD system would have the potential to benefit designers during conceptual designing more than current CAAD systems.

6. ACKNOWLEDGMENTS

This research is supported by an Overseas Postgraduate Research Scholarship and by a grant from the Australian Research Council. We wish to thank Dr Masaki Suwa for making the data available and Rob Saunders for discussing some ideas in this paper.

7. REFERENCES

Co, F.D. and K.W. Forster, 1998, *Frank O. Gehry: The Complete Works*. The Monacelli Press, Inc., New York, USA.

Ericsson, K.A. and H.A. Simon, 1993, *Protocol Analysis: Verbal Reports as Data*. MIT Press, Cambridge, MA.

Galle, P., 1999, "Design as intentional action: a conceptual analysis", *Design Studies*, 20 (1), p. 57-81.

Gero, J.S., (ed) 2000, *Artificial Intelligence in Design' 00*, Kluwer Academic Publishers, Dordrecht, Netherlands.

Goel, V., 1999, "Cognitive roles of ill-structured representations in preliminary design", in: J. S. Gero and B. Tversky (eds.) Visual and spacial reasoning in design, MIT, Cambridge, KCDCC, University of Sydney, Australia, p. 131-144.

Gross, M.D., 1996, "The electronic cocktail napkin - a computational environment for working with design diagrams", *Design Studies,* 17, p. 53-69.

McNeill, T., J.S. Gero and J. Warren, 1998, "Understanding conceptual electronic design using protocol analysis", *Research in Engineering Design* 10, p. 129-140.

Norman, D.A., 1998, *The Invisible Computer.* MIT Press, London, England.

Schön, D.A. and G. Wiggins, 1992, "Kind of seeing and their functions in designing", *Design Studies,* 13 (2), p. 135-156.

Suwa, M., J.S. Gero and T. Purcell, 2000, "Unexpected discoveries and S-invention of design requirements: important vehicles for a design process", *Design Studies,* 21, p. 539-567.

Suwa, M. and B. Tversky, 1997, "What do architects and students percive in their design sketches? A protocol analysis", *Design Studies,* 18, p. 385-403.

Tang, H.-H. and J. Gero, 2000, "Content-oriented coding scheme for protocol analysis and computer-aided architectural design", in: B.-K. Tang, M. Tan and Y.-C. Wong (eds) *CAADRIA 2000*, CASA, Singapore, p. 265-275.

Ullman, D.G., S. Wood and D. Craig, 1990, "The importance of drawing in the mechanical design process", *Computers and Graphics,* 14 (2), p. 263-274.

Knowledge management by information mining

Özer Ciftcioglu and Sanja Durmisevic
Delft University of Technology

Key words: Knowledge management, Information mining, Sensitivity analysis

Abstract: Novel information mining method dealing with soft computing is described. By this method, in the first step, receptive fields of design information are identified so that connections among various design aspects are structured. By means of this, complex relationships among various design aspects are modeled with a paradigm, which is non-parametric and generic. In the second step, the structured connections between various pairs of aspects are graded according to the relevancy to each other. This is accomplished by means of *sensitivity analysis,* which is a computational tool operating on the model established and based on a concept measuring the degree of dependencies between pairs of quantities. The degree of relationships among various design aspects so determined enables one to select the most important independent aspects in the context of design or decision-making process. The paper deals with the description of the method and presents an architectural case study where numerical and as well as non-numerical (linguistic) design information are treated together, demonstrating a ranked or elective information employment which can be of great value for possible design intervention during reconstruction.

1. INTRODUCTION

In architectural design process, one has to establish certain relations among the design information in advance to make design with sound rationales behind. The main difficulty at this point is that such relationships may not be determined because of various reasons. One example may be the vagueness of the architectural design data due to linguistic qualities in them. Another example may be the vaguely defined design qualities, which should be gradually fixed during the actual implementation, in order to maintain the flexibility of the design for architectural real-time decision-makings. To deal

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 533–545.
© 2001 *Kluwer Academic Publishers.*

with such flexible design information is not an easy task since the majority of the existing architectural design aids, so-called decision support systems, are based on concrete input design information to be provided and well-defined final goals. Here the problem is not only the initial fuzziness of the information but also the desired relevancy determination among all pieces of information given. Basically, if there is no relevancy between two aspects, they need not to be considered in that context of design. Acquiring this knowledge in advance might save considerable design efforts in the meanwhile. However presently, to determine the existence of such a relevancy is more or less a matter of architectural subjective judgement rather than a systematic non-subjective decision-making based on the existing design information. In this respect, the invocation of certain design tools dealing with such fuzzy information is essential for enhanced design decisions. Referring to the fuzziness of the information subject to treatment, naturally fuzzy logic tools are most appropriate to employ (Tanaka, 1997).

2. KNOWLEDGE MANAGEMENT

Knowledge is the formation of implicit and explicit restrictions placed upon objects, operations and relationships along with general and specific heuristics and inference procedures involved in the situation being modeled (Sowa, 1985). From the design activity viewpoint, *specific heuristics and inference procedures* play important role since there is no firm guidelines for a specific design task. Here, heuristic may be understood as any effort based on the performance of the case and intended for the accomplishment of the related task. In a design task, specific heuristic and inference procedures may play important role. This is especially the case if there are essential imprecision in the data and this is the case in this research. The imprecision in the data can be handled by fuzzy logic techniques so that generic information about the relevancy among the design items and the effectiveness of each design item in this context can be identified. This is a basic information mining procedure for the elicitation of information from the data. Information mining and knowledge management is essential activity in decision support systems with inference capability. Fuzzy expert systems can provide the required flexibility for dealing with the imprecise data and basically such a scheme is described in *figure 1*.

In a complex information environment, to establish the complete fuzzy rules dealing with the knowledge base is a formidable task. To alleviate the problem, the knowledge base can be formed in a distributed and structured form by means of learning (machine learning) so that the structure represents the fuzzy expert system altogether with consistent rules in any complexity.

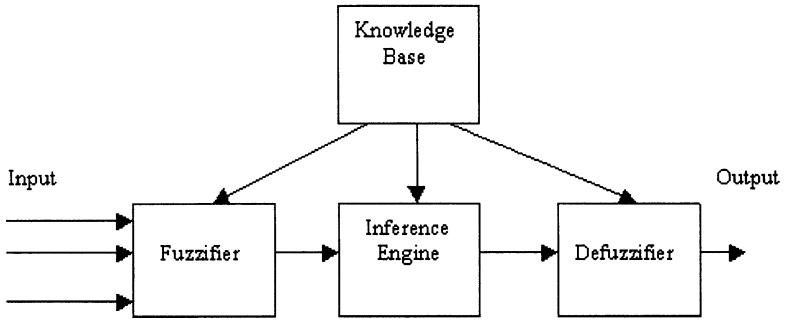


Figure 1. Scheme of fuzzy expert system

Here the main task is the establishment of the consistent rules. This can be achieved by performing information-mining methods with appropriate learning process, which should be specially designed for this purpose. Information mining is closely related to data mining. The main difference between information mining and conventional data reduction is the complexity of the data and the method of elicitation of the information being searched for. Data mining is in essence is the process of data reduction (Han, Kamber 2001). It is one of the dominant techniques of exploratory data analysis and there are a number of ways for it. Clustering is one of the essential methodologies of concern (Han, Kamber 2001; Krzysztof, Pedrycz, et al. 1998). It can broadly be divided into two categories as clustering with unsupervised and clustering with supervised learning. In the unsupervised case, given an amount of information, data set and the distance function, one starts with the number of clusters assigning each pattern to a separate cluster and proceeds to merge the clusters that are the closest. In the supervised case, we are concerned with a collection of labeled data that come in the form of ordered pairs. This can be seen as a feature vector describing the data and its class assignment. Eventually, the associations between the feature vectors and the class assignments are endeavored for determinations. Perhaps the major disadvantage of unsupervised clustering is due to fact that the clusters are established from the input data and the final associations with some output functions are performed afterwards. In this case the associations may not be optimal since the established clusters may not be the optimal representations with respect to associations being looked for. In the supervised case the clustering and associations are performed in a constructive way so that the most effective representation of associations being investigated is obtained. The accomplishment of this structure can be realized by means of a network, which has special knowledge management features suitable for dealing with imprecise data. Such a structure can be a radial basis function network (RBF) network. RBF networks form one of the

essential categories of neural networks. The main architectures, learning abilities, and applications are described in the literature (Brommhead, Lowe 1988; Moody, Darken 1989; Park, Sandberg 1991; Leonard, Kramer, et al. 1992; Chen, Manry 1993; Eleneyar, Shin 1994) where the learning and generalization abilities of these networks are outstanding. In particular, some interesting equivalence between RBF networks and fuzzy rule-based systems have been established (Jang, Sun 1993; Hunt, Haas, et al. 1998). Here, the antecedants (input premises) of the respective rules describing the fuzzy model give rise to a linguistic partition of the input space. An RBF network operates with fuzzy computational units, which are in essence cluster centers and they constitute the essential functional components in the structure. However, from information mining and knowledge management viewpoint, these cluster centers are called receptive fields to distinguish between information mining and data mining since selection of the receptive fields differs from conventional clustering techniques as explained below.

Referring to the form of the computational units in an RBF structure, especially their distribution in the input space is often critical for the performance of the network. In general, in the knowledge model by RBF networks, categorically two stages can be identified. In the first stage, encapsulation of some domain knowledge is carried out. In the second stage, a parametric learning process takes place. For the encapsulation of the domain knowledge there are two main sources of this knowledge:

- *A preliminary analysis of training data* where the data set is preliminarily analyzed and the receptive fields are determined. For this, general data clustering methods are of particular interest (Bezdek 1981).
- *Designer-oriented data analysis* where receptive fields are formed based upon intended design goals of designer. In this respect, the preceding category concerns self-organization and this category belongs to supervised organization. In this second stage, the domain knowledge can be more explicitly emphasized in the model. This allows focusing attention on the machine learning on some essential regions of the input space. As a result of this a multiresolutional character in knowledge modeling together with efficient learning is exercised.

Concerning information mining and knowledge management by RBF networks information is embedded locally in the form of a database where elicitation of information from the data requires special methods and techniques. In particular, referring to the encapsulation of the knowledge, the receptive fields are formed by means of second categorical source of information described above, where special type of supervised clustering for determination of the receptive fields is performed, as a first step. A parametric learning follows this, as a second step. These two steps in knowledge modeling are imperatively performed by orthogonal least squares

(OLS) algorithm (Chen, Cowan, et al., 1991). In particular, by conventional clustering, cluster centers are located anywhere in the input space, matching the clustering process to some prescribed clustering criteria. This is purely a mathematical treatment and the centers identified might have no correspondence to a physical entity or reality. However, for knowledge management, the model for knowledge base should be constructed on actual data rather than some mathematical abstractions derived from data. In this respect, in the OLS, the centers are part of the data as receptive fields rather than clusters. This means, each receptive field is a set of data in the form of a data vector in a multidimensional input space. More explicitly, they are a part of the data reflecting the exact information present in the data to the knowledge model without any mathematical abstraction. They refer to the appropriate central locations in the model according to data at hand. By doing so, the domain knowledge is effectively emphasized by means of two important gains accomplished. In the first place, the effective management of knowledge by the appropriate distribution of the receptive fields is carried out. In the second place, the enhanced generalization capability of the knowledge model is guaranteed by selecting appropriate number of receptive fields in the model. In general, the selection of number of clusters or receptive fields in such a model is critical and therefore its appropriate selection is essential concern. In the literature, this phenomenon is referred to as *bias-variance dilemma* (Duda, Peter, et al., 2001) and mostly treated in the context of neural networks (Haykin, 2000). This phenomenon plays also important role on the model developed in this research. The dilemma basically manifests itself by the estimations through the model. If the number of receptive fields in the model are excessive, than the model errors with the training data are relatively small while the same errors are relatively high for unknown data applied to the input of the model, and vice versa. Since overall effect on the quality of the knowledge model is the combination of these two conflicting errors, the model should have appropriate number of receptive fields as optimal (see *figure 2*). The model error obtained from the knowledge model with unknown data at its input is used as a measure for performance of the model and it is referred to as *generalization error.* In this context, the performance of the model is expressed in terms of its *generalization capability* measured by the generalization error.

3. KNOWLEDGE MANAGEMENT FOR ARCHITECTURAL DESIGN - CASE STUDY

In this research, knowledge management by information mining is performed for an architectural study related to underground station design. The data collected comprised various design features, in general. The knowledge management considers certain aspects of the station design. For this purpose the dependency among various design features is required. This dependency can be conceived as an input and output structure where the inputs are mapped to the outputs via a knowledge model. Therefore, from the data at hand the general dependency information was needed to devise a model in a structural and continuous form. Such a model could be used for any set of design requirements as a knowledge base where for these requirements at the input, it provides consistent design solutions at the output.

In basic mathematical terms, if we assume that the input vector x(t) represents a n-dimensional vector of real-valued fuzzy membership grades: $x(t) \in [0,1]^n$, then, the output y(t) of the model represents an m-dimensional vector of corresponding real-valued membership grades, $y(t) \in [0,1]^m$. This structure actually performs a nonlinear mapping from an n-dimensional hypercube $I^n=[0,1]^n$ to an m-dimensional hypercube $I^m=[0,1]^m$:

$$x(t) \in [0,1]^n \rightarrow y(t) \in [0,1]^m$$

In the actual implementation, $x_i(t)$ $(i=1,2,\ldots,n)$ in x(t) represents the degree to which an input fuzzy variable $x_i'(t)$ belongs to a fuzzy set, while $y_j(t)$ $(j=1,2,\ldots,m)$ in y(t) represents a degree to which an output fuzzy variable $y_j'(t)$ belongs to a fuzzy set. Further in the text, a case study is presented and based on that data sensitivity analysis is conducted.

A data set obtained for Blaak underground station in the Netherlands was used for training as a case study. This is an important exchange station, which is situated in the center of Rotterdam. It is at the same time a tram, metro and a train station. Tram station is situated on a ground level. Metro platforms are one level below ground (at approximately -7m) and train platforms are two levels below ground (at approximately -14 meters). From 27th May till 30th May 2000, one thousand of questionnaires were handed out to the passengers visiting the station. The questionnaire covered aspects that are related to safety and comfort at the station. In total there were 43 aspects in input space each having five possible options and two aspects in the output space again having five possible options. The latter two are design variables being *safety* and *comfort*. The input aspects, which are identified to be related to comfort are given in *table 1* and those related to safety are

presented in *table 2* (Ciftcioglu, Durmisevic, et al., 2001). Main purpose of the questionnaire was to provide information on user's perception regarding specific spatial characteristics of that station. The questions covered all aspects given in *table 1* and *table 2* and additional two final questions were related to users' perception of public safety and comfort at Blaak station.

In general, the way in which data is measured is called a level of measurement or the scale of measurement for variables. The organization of variables determines how data can be analyzed (McGrew, Monroe, 1993). There are four kinds of scales in which variables can be measured: the *nominal* scale, the *ordinal* scale, the *interval* scale and the *ratio* scale (Dalen, Leede, 2000). In this research we have used an *ordinal scale measurement*, where such measurement involves placement of values in a rank order, in order to create an ordinal scale variable. The relationship between observations takes on a form of 'greater than 'and 'less than'. In the questionnaire, variables are divided in a five-point scale. The use of a three-point scale or a five-point scale in a questionnaire are the most common (Baarda , de Goede, 1997). For this questionnaire a five-point scale was used in order to generate a distinct approach. In such way, respondents have the opportunity to 'strongly agree' or 'strongly disagree' with a question or to strongly express their opinion regarding design issues.

Table 1. Aspects related to comfort (total 28)

Attractiveness	**Wayfinding**	**Daylight**	**Physiological**
Colour	To the station	Pleasantness	Noise
Material	In station	Orientation	Temperature winter
Spatial proportions	Placement signs		Temperature summer
Furniture	Number of signs		Draft entrance
Maintenance			Draft platforms
Spaciousness entrance			Draft exchange areas
Spaciousness train platform			Ventilation entrance
Spaciousness metro platform			Ventilation platforms
Platform length			
Platform width			
Platform height			
Pleasantness entrance			
Pleasantness train platform			
Pleasantness metro platform			

Table 2. Aspects related to safety (total 15)

Overview	**Escape**	**Lighting**	**Presence of people**	**Safety surr.**
Entrance	Possibilities	Entrance	Public control	Safety in surrounding
Train platform	Distances	Train platform	Few people daytime	
Metro platform		Metro platform	Few people night	
Exchange area		Exchange area		
		Dark areas		

In a 6 weeks period 219 completed questionnaires were returned. Some cases were immediately excluded since they failed the control question, which was in the questionnaire in order to improve reliability of the outcomes. For training we used in total 196 cases and 7 cases were used in order to test the network performance.

Figure 2 provides the outcomes of the network training. *Figure 3* provides validation of the network performance for both comfort and safety aspects. For training 196 cases were used and for the knowledge model validation, the number of receptive fields selected was 90 out of graded sequence of centers which was in total 196. The dimension of the input space was 43. The network estimation is rather reasonable considering the fact that there are a number of possible combinations in the input space that can be represented as input information. More precisely at the order of 5^{43} different combinations are possible, since each dimension has five options, referring to earlier mentioned questionnaire. Thanks to the information mining in the form of receptive field clustering, in such a large dimensional input space, relatively small number of convergence points (altogether 90 receptive fields out of 196 input sets of data) are precisely located. It is important to point out that, the number of receptive fields is found to be optimal around 90 for this particular model and this is related to the bias variance dilemma mentioned before.

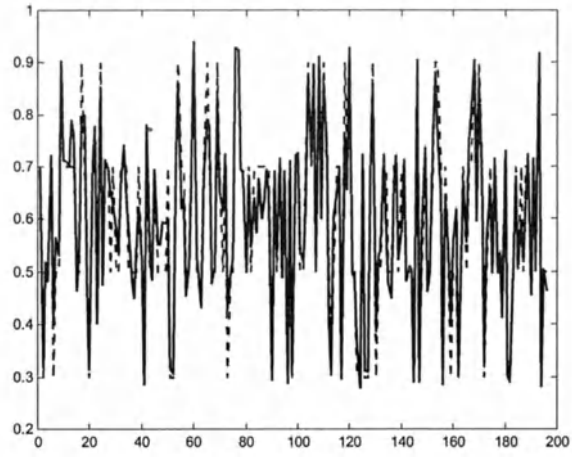


Figure 2. Network training for 196 cases with 43 inputs and 1 output as *comfort* variable with 90 receptive fields. Broken lines represent the knowledge model response to the training data after training. Continuous lines represent the actual knowledge used for modeling. Some difference is visible and referring to the bias-variance dilemma, it has positive effect for the generalization capability of the model (see Figure 3)

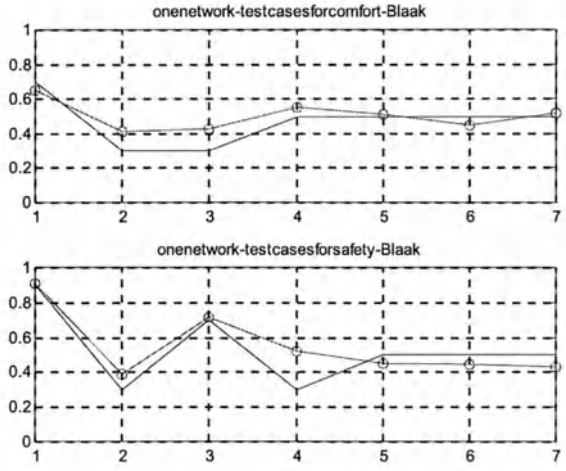


Figure 3. Test results of the network performance for comfort (upper picture) and safety (lower picture). Line with circles represents the estimated value by the network, while the other line is the actual value

In *figure 2*, the broken lines represent the knowledge model response to the training data after training. The difference between the actual data and

model response is basically modeling error, which is deliberately allowed to enhance the generalization capability of the model.

It is interesting to note that in *figure 2*, for the number of receptive fields equal to the number of input sets of data, which is 196 in this case, the modeling error virtually vanishes at the price of large generalization errors.

Having trained the network, whole information is stored in a compact way, which makes further knowledge management more efficient. It is also important to mention that it is necessary to train the network including aspects for comfort and safety at the same time, rather than to separate and train as two independent networks. The reason for that is rather obvious, since comfort and safety are very much related to each other, and certain issues related to one are embedded in the other as well and therefore would be rather superficial to separate them as independent information in the network training. This was confirmed through experiments as well (Durmisevic, Ciftcioglu, et al., 2001).

The relative dependency of the input variables on comfort and safety is identified by means of sensitivity analysis (Saltelli, Chan, et al. 2000) where basically the gradients of comfort and safety with respect to each variable in the input space is computed. The results are shown in *figure 4*, where the '*x*' axis represents 43 dimensions in the input space, and the '*y*' axis is a sensitivity of that dimension to the comfort and safety at the output, ranked on a scale from 0 to 1. From this figure, In *table 3* and *table 4*, only some most sensitive aspects are mentioned as indication of sensitivity analysis results. Since the computation is possible with analytical computation accurate results are obtained.

Table 3. Order of aspects that are most sensitive to feeling of comfort at Blaak station

Number	relative importance value	Sensitivity to comfort
1	1.0000	Spaciousness metro platform
2	0.8972	Spatial proportions
3	0.8723	Platform width
4	0.8171	Pleasantness metro platform
5	0.7647	Platform height
6	0.7403	Spaciousness entrance
7	0.6738	Pleasantness train platform
8	0.6508	Lighting of train platform
9	0.6102	Platform length

Table 4. Order of aspects that are most sensitive to feeling of safety at Blaak station

Number	relative importance value	sensitivity to safety
1	1.0000	Safety in surrounding
2	0.8202	Few people present during night
3	0.7266	Few people present during daytime
4	0.3680	Pleasantness of metro platform

Number	relative importance value	sensitivity to safety
5	0.3517	Lighting of train platform
6	0.3239	Wayfinding in station

On one hand, since the first 28 variables are directly related to comfort, the associations of these variables to comfort are prevailing in that region (*see table 1 and figure 4 upper picture for comparison*). On the other hand, since the last 15 variables are directly related to safety, the associations of these variables to safety are clearly prevailing in that region (*see table 2 and figure 4 lower picture for comparison*). Yet it is noticeable that certain aspects for safety are very much influencing comfort and other way round. One example is an aspect 36 being 'lighting at the train platform'. When we compare two sensitivity analysis results for comfort and safety we notice that this aspect is very sensitive to comfort, even though in the first place it was assumed that it might be more related to safety (*see table 2*). In that sense, sensitivity analysis proved to be an effective method indicating validity/invalidity of assumptions made. This provides additional confidence on the quality of the knowledge management and the validity of the knowledge model developed.

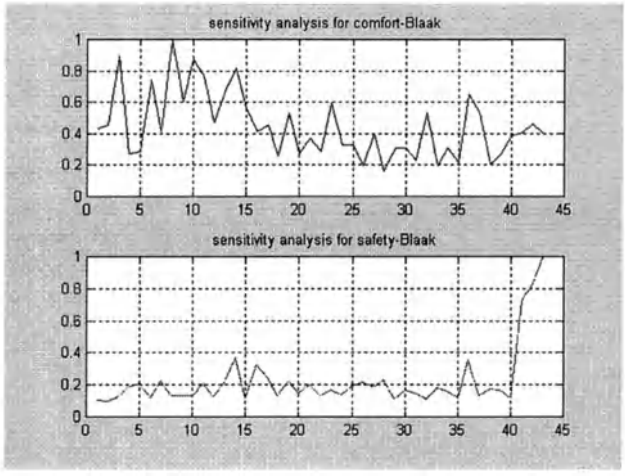


Figure 4. Upper figure is the sensitivity of comfort to input space variables and lower figure is the sensitivity of safety to the input space variables

The following table (*table 5*) gives an insight into final outcomes of estimation of comfort and safety provided by the passengers at Blaak station.

Table 5. [Evaluation of comfort and safety aspects for the Blaak station expressed in %]

5 scale measurement	Comfort	Safety	5 scale measurement
Very uncomfortable	1.4	3.7	Very unsafe
Uncomfortable	15.9	16.4	Unsafe
Reasonably comfortable	35.5	35.5	Reasonably safe
Comfortable	36.4	30.8	Safe
Very comfortable	10.7	13.6	Very safe

The outcomes from the knowledge model indicated outstanding potentiality of the model for gaining detailed insight into the information at hand and the effective use of such design information in a structural form as a knowledge base, for enhanced architectural design decisions. In the context of knowledge management, this knowledge model was especially designed to assess the qualitative aspects of design, including technical and quantitative values. The results of this study are valuable for eventual reconstruction of Blaak station, since it provides an architect with a valuable information regarding different aspects considered in this research. Numerous conclusions can be derived out of this model, like for example, the fact that the safety in the surrounding is the most determining aspect for feeling safe in the station. This means that no matter which architectural intervention are made inside the station, if a designer does not consider the surrounding where this station is situated and possible improvements in that surrounding, it is less likely that people will feel safer at that station.

4. CONCLUSIONS

Design knowledge management by information mining was described. For this purpose, the presentation revealed a development in knowledge acquisition and modeling using *machine learning* methods. In a complex information environment, for specific tasks such as a specialized design task (underground building design, in the present case) with a rich set of design information, a systematic elicitation and account of information in the form of a relational model is necessary. By means of this, consistent design is achieved which matches predefined design criteria. With an actual case study, the research indicated the effectiveness of the present information mining and knowledge management approach in building design. Eventually, this implies the effectiveness of the method as an approach for knowledge-enhanced design. Since the structured information presents a model, which is based on design information and implemented by machine learning, the knowledge base so formed becomes free from subjective decision-makings.

5. REFERENCES

Baarda, D.B. and Goede, de M.P.M. , 1997, *Basisboek Methoden en Technieken,* Educatieve Partners Nederland, BV, Houten

Bezdek C. (1981), *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum, New York

Broomhead, D.S. and Lowe, D., 1988, "Multivariable Function Interpolation and Adaptive Networks", *Complex Systems, 21, p.321-355*

Dalen, van J. and Leede, de E. , 2000, *Statistisch Onderzoek met SPSS for Windows*, Uitgeverij Lemma BV, Utrecht

Duda R.O., Peter E.H. and David G. Stork, 2001, *Pattern Classification*, Wiley Interscience, New York

Durmisevic, S., Ciftcioglu, Ö. and Sariyildiz, S., 2001, "Knowledge Modeling by Artificial Intelligence for Underground Space Environment", *Conference Proceedings EuropIA8, 25-27 April 2001, Delft, The Netherlands*

Elenayar S. and Shin Y.C., 1994, "Radial Basis Function Neural Network for Approximation and Estimation of Nonlinear Stochastic Dynamic Systems", *IEEE Trans. Neural Networks, Vol.5, pp.594-603*

Chen S, C.F.N. Cowan and Grant, P.M., 1991, "Orthogonal Least Squares Algorithm for Radial Basis Function Networks", *IEEE Trans. Neural Networks, Vol.2, No.2, March*

Chen M.S. and Manry M.T., 1993," Conventional Modeling of the Multilayer Perceptron Using Polynomial Basis Functions", *IEEE Trans. Neural Networks, Vol.4, pp.164-166*

Ciftcioglu, Ö., Durmisevic, S. and Sariyildiz, S., 2001, "Multi-Resolutional Knowledge Representation", *Conference Proceedings of EuropIA8, 25-27 April 2001, Delft, The Netherlands*

Han J. and Kamber M., 2001, *Data Mining: concepts and techniques*, Morgan Kaufmann, San Francisco

Haykin S., 2000, *Neural Networks:a comprehensive foundation*, Prentice-Hall, Upper Saddle River

Hunt K.J., Haas R. and Murray-Smith R., 1998, "Extending the Functional Equivalence of Radial Basis Function Networks and Fuzzy Inference Systems", *IEEE Trans. Neural Networks, Vol.7, No.3, May*

Jang J.S.R. and Sun C.T., 1993, "Functional Equivalence Between Radial Basis Function Networks and Fuzzy Inference Systems", *IEEE Trans. Neural Networks, Vol.4, No 1,*

Krzysztof J.C, Pedrycz W. and Swiniarski R.W., 1998, *Data Mining Methods for Knowledge Discovery*, Kluwer Academic, Boston

Leonard J.A., Kramer M.A., and Ungar L.H., 1992. "Using Radial Basis Functions to Approximate a Function and Its Bounds", *IEEE Trans. Neural networks, Vol.3*

McGrew, C. J. & Monroe C. B. 1993, *An Introduction to Statistical Problem Solving in Geography*, Wm. C. Brown Communications, Inc., Dubuque

Moody J. and Darken C., 1988, "Fast Learning with Localized Receptive Fields", in Proc. Connectionist Models Summer School, D. Tourezky, G. Hinton, and T.Sejnowski (eds.), Carnegie Mellon University, Morgan Kaufmann Publishers

Park J. and Sandberg I.W., 1991. "Universal Approximation Using Radial Basis Function Network", *Neural Computation, Vol.3, pp.246-257*

Saltelli A., Chan K. and Scott E. M., (Eds.), 2000. *Sensitivity Analysis*, Chichester:Wiley

Sowa, J. F., 1985, *Conceptual Structures*. Reading, MA: Addison-Wesley

Tanaka K. ,1997, *An Introduction to Fuzzy Logic for Practical Applications*, Springer, Berlin

The Topics of CAAD

A Machine's Perspective

Ziga Turk, Tomo Cerovsek, Bob Martens
University of Ljubljana, Slovenia.
Vienna University of Technology, Austria.

Key words: CAAD, machine learning, clustering, pattern recognition

Abstract: Ontology of a scientific field typically includes a taxonomy that breaks up the field into several topics. The break-up is present in the organisation of information in books, libraries and on the Web. An on-line database of papers related to CAAD called CUMINCAD was created and it includes over 3000 papers with abstracts. They are available through the search interface - one knows an author or a keyword and can find the papers where such keyword or author's name appears. Alternative interface would be through browsing papers topic by topic. The papers, however, are not categorised. In this paper, we present the efforts to use the machine learning and data mining techniques to automatically group the papers into clusters and create a set of keywords that would label a cluster. The hypothesis was that an algorithm would create clusters of papers automatically and that the clusters would be similar to the groupings a human would have made. We investigated several algorithms for doing an analysis like that but were unable to prove the original hypothesis. We conclude that it requires more than objective statistical analysis of the words in abstracts to create an ontology of CAAD.

1. INTRODUCTION

An important attribute of any scientific discipline are its framework, vocabulary, methodology, paradigms and structure - as Kuhn (1962) calls them, the *"conceptual boxes ... into which the scientists, by a rather a strenuous and devoted attempt to force nature into"*. According to Seni and Hodges (1997), a field of science is actually defined with the following attributes:

- Axiology defines a value system in the field.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 547–560.
© 2001 *Kluwer Academic Publishers.*

- Ontology defines "what exists" and (formally) specifies the conceptualisation of the field.
- Epistemology specifies what constitutes appropriate knowledge in the field, where is it and how it can be represented and transferred.
- Methodology specifies the appropriate rules of inquiry.

Ontology typically includes taxonomy or some other way of breaking up the field into several topics. Each scientist involved in a field, such as CAAD, could sketch such taxonomy. Implicitly, it is present in the organisations of conference sessions, textbook sections, keywords systems used to tag scientific papers and in the portals on the World Wide Web. While the firsts uses may be scientifically intriguing and could provide some introspective into the CAAD community, our practical interest lies in the latter.

1.1 Motivation

A comprehensive database of CAAD papers called CUMINCAD was created (Martens and Turk, 1999) on the World Wide Web (http://itc.fgg.uni-lj.si/cumincad/). Currently it includes over 3000 papers. Each paper includes all vital bibliographic information such as title, authors, publication data etc. (*Figure 1*). Full abstracts are available as well. Some 500 papers from the eCAADe conferences include full texts in .pdf format.

Currently the papers are available through the search interface: if one knows an author or a keyword or any word or words that appear in the abstract, one can find the papers, where the search term appears.

Searching, however, implies that the user knows what she is looking for. Another approach to access the data is through browsing a structured collection of records. The structure should tell the user what items are similar, which are different, and how. The simplest structures of this kind are clusters or groups of similar data items. These groups can be ordered into a hierarchy. A well-known example of hierarchical interface to the content of the Web is the portal Yahoo. The groups may be predefined as classes. In this case we talk about classification and not about clustering.

The users of CUMINAD would like to (1) browse through the papers by the topics of CAAD, or (2), given a paper or a group of papers, find the papers that are similar to those. Because of the size of the database and the fact that the work has been done on a shoestring budget, manual classification of the papers has been ruled out.


CUMINCAD
Paper 0c9c:
Prescribing Designs

authors Tweed, Christopher
year 1999
title Prescribing Designs
source Architectural Computing from Turing to 2000 [eCAADe Conference Proceedings / ISBN 0-9523687-5-7] Liverpool (UK) 15-17 September 1999, pp. 51-57
summary Much of the debate and argument among CAAD researchers has turned on the degree to which CAAD systems limit the ways in which designers can express themselves. By defining representations for design objects and design functions, systems determine what it is possible to describe. Aart Bijl used the term 'prescriptiveness' to refer to this property of systems, and the need to overcome it was a major preoccupation of research at EdCAAD during the 1980s, including the development of the MOLE (Modeling Objects with Logic Expressions) system. But in trying to offer designers the freedom that was judged to be essential to evolving design practices, MOLE transferred much of the burden of programming from system developers to end-users - you can have any design objects you want, as long as you write the code. Close examination of MOLE's logic reveals that it too had to rely on fundamental definitions that, even if not domain-specific, are certainly historically contingent. This paper will return to the issue of prescriptiveness, summarising the lessons learned from the MOLE 'experiment,' and identifying new prescriptions that are deciding what designs can be. Looking beyond computer representations, we find that designs are shaped by much larger, and arguably more powerful, historical, social and cultural forces surrounding design practice. These forces are shaping the way CAAD is used and how new systems are conceived and developed.
keywords Bijl, Prescriptiveness
series eCAADe
email c.tweed@qub.ac.uk
full text Full text not available

Figure 1. Information about one paper in CUMINCAD

CUMINCAD database is handled by a Web database tool Woda (Turk, 1998) and the goal of the first author is also to create a generic solution that could be applied to any database handled by Woda.

Ideally we would use machine learning and data mining techniques to automatically group the papers into clusters and create a set of keywords that would label a cluster. Based on such clusters, categories would be defined and the entire database would be organised accordingly.

1.2 Hypothesis

The hypothesis was that it is possible to write an algorithm that would create clusters of papers automatically and that the clusters would be similar to the human understanding of the field. If we name the similarity of papers p_1 and p_2 with $\sigma(p_1,p_2)$ than we believed that, given a paper p, we could find a set of similar papers $s_{[1..n]}$ for which is true that

$$\sigma(p, s_i) >> t$$

and another set $d_{[1..m]}$ where

$$\sigma(p, d_i) << t$$

where t is some threshold value. Depicted graphically, the similarity of a given paper to other papers should look as shown in *Figure 2*.

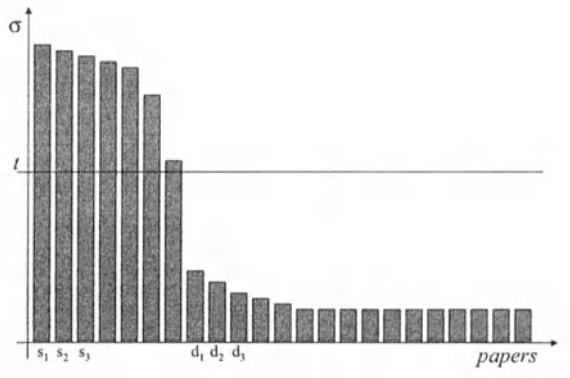


Figure 2. Desired similarity between papers.

Set of papers *s* would belong to the same cluster because the similarity is much higher that a threshold value. All other would be in set *d* where the similarity is much lower.

1.3 Paper structure

In Section "Methodology" we first present the related work then the various algorithms used to computer the similarity of papers, which is the basis for categorisation and clustering. In Section "Analysis of the CUMINCAD data" we present the results of the analysis of the CUMINCAD dataset. In Section "Conclusions" we evaluate how well did the analysis meet the stated hypothesis.

2. METHODOLOGY

2.1 Related work

Various AI technologies for analysing text databases are known since the late 1970s (van Rijsbergen 1979). They became particularly popular after the explosion of the World Wide Web, when the search engines were looking for the technologies to increase the relevance of the searches (Zamir and Etizioni, 1999) or build some intelligence into Web browsing (Mladenic, 1999). An example of an "intelligent" interface to bibliographic data is for example the www.researchindex.com. Several Websites implement such technology, for example AltaVista and Google (Bring and Lawrence, 1998).

In the area of architecture, engineering and construction not much work has been done in this direction, particularly because engineering information is not predominantly text, but drawings and product data. Maher and Simoff

(1998) applied those techniques to searching archives of old projects and to discover new knowledge out of them. Christiansson (1998) used them to simplify access to knowledge bases. The importance of data mining technologies was also acknowledged by Hovestad (1993) in an attempt to manage unstructured design data.

2.2 Algorithms

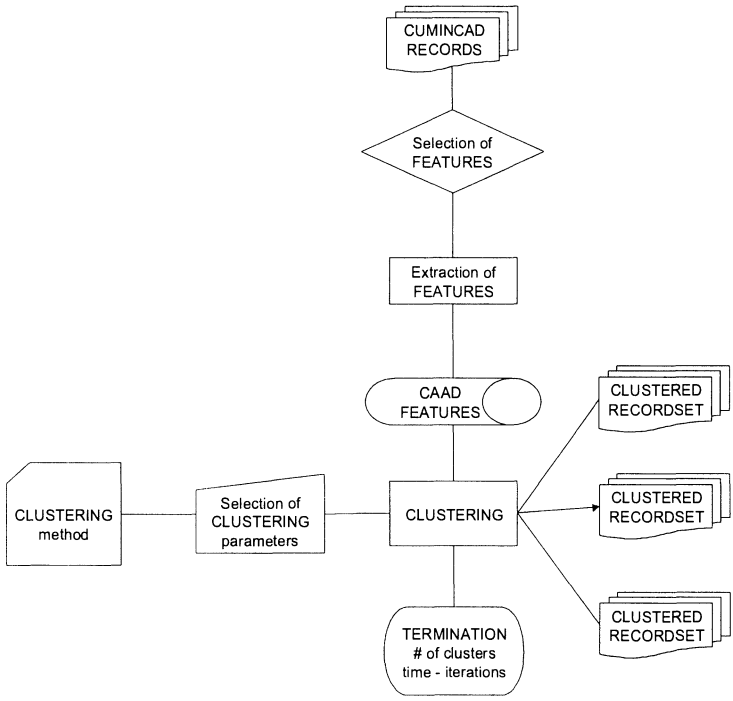


Figure 3. Document clustering process

In the 1970s and 1980s several algorithms analysing text, comparing documents and extracting most important phrases or sentences out of the text were developed (overview in Willett, 1988, and Zamir and Etzioni 1998). Quite a few programs were written that are freely available.

Both classifications of documents and clustering rely on being able to compute similarities between two documents. The overall procedure (see *Figure 5*) consists of the following steps: (1) selection of features, (2) extraction of features, (3) vectorisation of documents (4) computing similarities between documents and finally (5) clustering or classification itself.

2.2.1 Selection of features

The parts of the documents or data records considered relevant are selected. For example in the case of our study of bibliographic information, we selected only the title, abstract and keywords and omitted the author and publication information.

2.2.2 Extraction of features

Relevant data is extracted out of the selected data and converted into a format suitable for further processing. Typical first steps in the extraction include the elimination of stop words and stemming. Stop words are words that appear too many times to be of relevance, or words that are known not to be able to contribute to the computation of similarities between records. For example both "the" and "cad" are very frequent. The first would appear in the available stop-word libraries for the English language. The second was found very frequently in the CUMINCAD database.

Stemming is a procedure that discovers roots of the terms and makes sure that for example "mouse" and "mice", "house" and "houses" are treated as a single term.

2.2.3 Vectorisation of documents

This step is usually described as a part of similarity computation, however, in the section Conclusions we explain why we find it important, to discuss it outside that scope. Documents need to be converted to a vector of numbers. The simplest form is to create a **term frequency matrix** [ωr] where the rows are records and columns the terms.

$$[\omega r] = \begin{bmatrix} \omega r_{11} & \omega r_{12} & \omega r_{13} & . & . & . & \omega r_{1T} \\ . & . & & & & & . \\ . & & & & & & . \\ \omega r_{N1} & \omega r_{N2} & . & . & . & . & \omega r_{NT} \end{bmatrix}$$

T is total number of different terms in all records and N is the number of records. ωr_{12} tells us, how many times does term number 2 appear in record 1.

2.2.4 Computing similarities between documents

The algorithms for the computing of similarities between documents differ in how they calculate the weight they give to words appearing in a document. One approach might only count if a

word is present or not; another, how many times the word is repeated and how many words are there in total etc. etc. Perhaps the most often used approach is the naive Bayes. As an illustration, we present the CYBERMAP (Salton, 1989).

2.2.5 Document clustering

There are two types of document clustering algorithms. The bottom-up algorithms start with each document as one cluster and then group them to form bigger clusters of documents. Top down algorithms work top down, iteratively, so that they create two groups of documents that are as different to each other as possible, than each of the groups is further split into two etc. Clustering can result in a linear or hierarchical set of clusters. For a broader overview see Willett (1988).

3. ANALYSIS OF THE CUMINCAD DATA

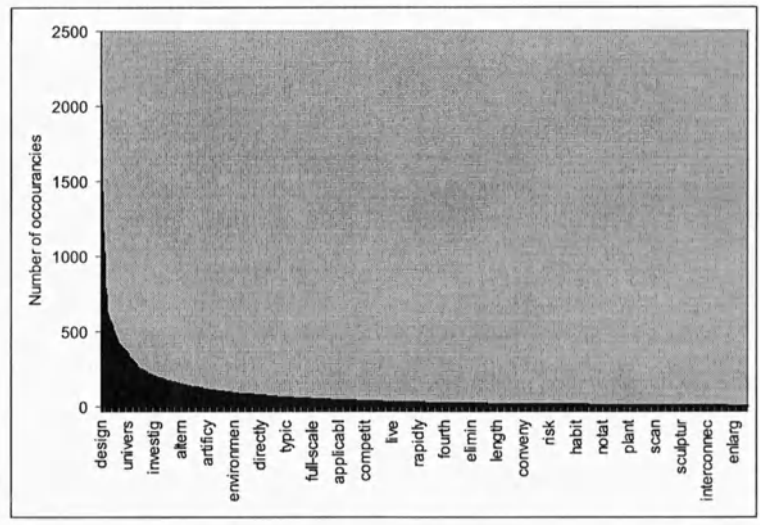


Figure 4. Occurrences of some words from whole collection

Table 1: Collection statistics (terms related statistics in general and vector specific)

Number of all records (in whole collection)	Number of stem terms (in whole collection)	Average term usage (in whole collection)	Average term vector length	Standard deviation of vector length	Average frequency of terms in vector	% of terms with freq. 1
3042	11762	14.76	57.39	30.32	1.39	78.66

To enhance CUMINCAD, the described procedures will be used on the entire 3000 papers. To compare various approaches and parameters we extracted only the 103 papers from the eCAADe 1999 conference in Liverpool. The structure of the proceedings, done by the organisers, provides a chance to compare the results of the machine made clusters to those made by a human editor of the proceedings. *Figure 4* illustrates the most frequent words. Detailed statistical analysis of the whole collection is summarized in Table 1.

3.1 Similarity measures

Figure 5 shows part of the similarity matrix of the Liverpool data set.

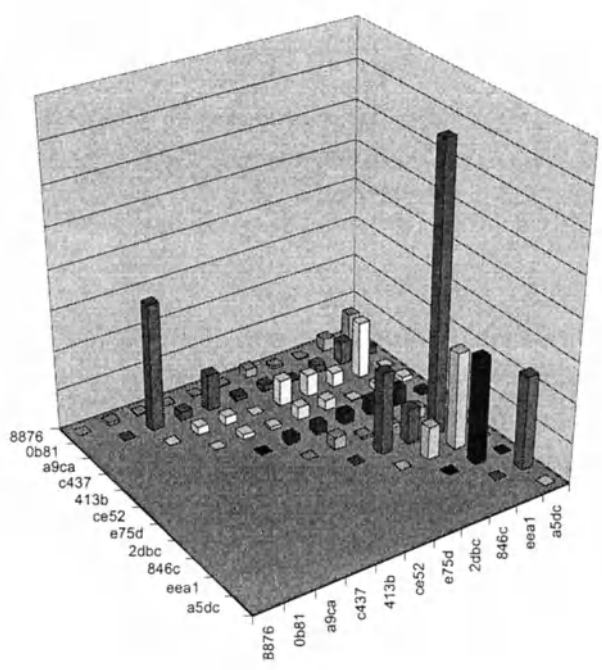


Figure 5. Visualisation of similarity matrix based on the Liverpool eCAADe papers. If a row for one given paper would be extracted and sorted a threshold value of as in Figure 2 would not be evident (Figure 6).

On the other hand, the graph of records most similar to Tweed (1999) is depicted in Figure 6. This is quite unlike the graph in the hypothesis. If there is no clear boundary for a single record, can we hope to find clusters with any degree of relevance?

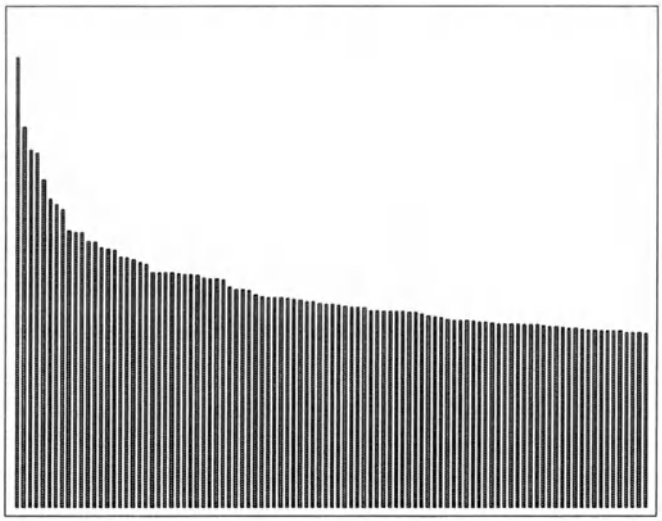


Figure 6. Graph of records most similar to Tweed(1999).

3.2 Clustering with Clustifier

Developed at NRC Canada, Clustifier (Scott 1998) uses representation based on semantic as well as syntactic linguistic knowledge. The results of the clustering are shown in *Figure* and *Figure 7.*

Cluster 0: caad education, computer aided architectural design, experience, caad research, design process, techniques, world wide web, presents

Cluster 1: spatial, methods, teaching, architectural education

Cluster 2: architecture, learning, virtual reality, experience, architectural design, based design, built environment, design process, design studio, caad

Cluster 3: virtual design studios, design process, virtual reality, approach, collaborative design

Cluster 4: information technology, urban planning, research projects, techniques, architects, visualisation, 1998, presents, analysis, computer technology

Cluster 5: computer aided architectural design, architectural computing, models, computer graphics, architectural representation, analysis, design process, techniques, year 2000

Cluster 6: computer aided architectural design, learning, information technology

Cluster 7: 86 architectural education, virtual environments, architectural design education, modelling, information technology, computer aided architectural

Figure .: Clusters titles as recognised by Clustifier. Note that this tool identifies phrases!

Cluster 0: caad education, computer aided architectural design, experience, caad research, design process, techniques, world wide web, presents

(1) *Af Klercker, Jonas*: **CAAD - Integrated with the First Steps into Architecture** ; (2) *Koutamanis, A., Barendse, P.B74 and Kempenaar, J.W.*: **Web-based CAAD Instruction: The Delft Experience**; (3) *Papanikolaou Maria and Tuncer, Bige*: **The Fake.Space Experience - Exploring New Spaces**; (4) *Koutamanis, Alexander*: **Approaches to the Integration of CAAD Education in the Electronic Era: Two Value Systems**; (5) *Tweed, Christopher and Carabine, Brendan*: **CAAD in the Future Perfect**; (6) *Jakimowicz, Adam*: **An Intuitive CAAD**; (7) *Loy, Hollis A.*: **Foundation for a Thorough CAAD Education**;

Cluster 1: spatial, methods, teaching, architectural education

(1) *Voigt, A., Walchhofer, H.P. and Linzer, H.*: **The Historico-cultural Past as Spatial-related Cognition Archives: Computer-assisted Methods in the History of Urban Development, Archeology and History of Art**; (2) *Blaise, Jean-Yves and Dudek, Iwona*: **SOL: Spatial and Historical Web-Based Interface for On Line Architectural Documentation of Krakow's Rynek Gowny**; (3) *Moorhouse, Jon and Brown,Gary*: **Autonomous Spatial Redistribution for Cities**; (4) *Kosco, Igor*: **How the World Became Smaller**;

Cluster 2: architecture, learning, virtual reality, experience, architectural design, based design, built environment, design process, design studio, caad

(1) *Burry, Mark, Dawson, Tony and Woodbury, Robert*: **Learning about Architecture with the Computer, and Learning about the Computer in Architecture**; (2) *Zarnowiecka, Jadwiga C.*: **AI and Regional Architecture**; (3) *Coppola, Carlo* : **Computers and Creativity in Architecture** ; (4) *Benedetti, Cristina and Salvioni, Giulio*: **The Use of Renewable Resource in Architecture: New Teaching Methodologies**; (5) *Reffat, Rabee M. and Gero, John S.*: **Situatedness: A New Dimension for Learning Systems in Design**; (6) *Mortola, E., Giangrande, A., Mirabelli, P. and Fortuzzi, A.*: **Interactive Didactic Modules for On-Line Learning via Internet**; (7) *Gavin, Lesley*: **Architecture of the Virtual Place**; (8) *Porada, Mikhael*: **Virtual Analogy and Architecture**; (9) *Heylighen, Ann and Neuckermans, Herman*: **Learning from Experience: Promises, Problems and Side-effects of CBD in Architecture**; (10) *QaQish, Ra'Ed K.*: **Evaluation as a Key Tool to Bridge CAAD and Architecture Education**;

Figure 7. Some clusters as generated by the Clustifier program and related papers.

- comput architectur futur univers caad technologi studi educ applic method
  - comput caad build develop aid design cad work cultur region
    - programm artifici design intellig evolut space gener pattern algorithm investig
    - learn reflect research program institut lectur interfac histor base offer
  - educ technologi base univers web plan effect design level teach
    - evalu ict cours introductori wide approach emphasi electron barrier year
    - educ model theori architectur concern role direct specif lead word
- represent architectur present paper digit media environ approach model program
  - design analysi aspect learn situat knowledg concept obtain theori preliminari
    - project method work oper technologi discuss studio media plan educ
    - system model design form idea dynam analyt comput reason behaviour
  - virtual environ prototyp base model aim imag work includ basic
    - space student project effect exercis particip workshop develop consist vr
    - world virtual view comput build real time analogi model multi

Figure 8. Topic of CAAD as recognised by the Crossbow program. Characteristic terms are listed on the section headings. Note that the words are stemmed.

3.3 Clustering with Crossbow

Crossbow (McCallum 1996) is a classification tool that uses a hierarchical, top town, EM (expectation, maximisation) clustering technique. After over 150 iterations it produced the result as shown in *Figure 8* and *Figure 9*.

Cluster 1: learn reflect research program institut lectur interfac histor base offer

(1) *Blaise, Jean-Yves and Dudek, Iwona*: **SOL: Spatial and Historical Web-Based Interface for On Line Architectural Documentation of Krakow's Rynek Gowny**; (2) *Bridges, Alan*: **Progress? What Progress?**; (3) *Brown, A., Knight, M. and Berridge, P. (Eds.)*: **Architectural Computing from Turing to 2000 [Conference Proceedings]**; (4) *Burry, Mark, Dawson, Tony and Woodbury, Robert*: **Learning about Architecture with the Computer, and Learning about the Computer in Architecture**; (5) *Cha, Myung Yeol and Gero, John*: **Style Learning: Inductive Generalisation of Architectural Shape Patterns**; (6) *De Mesa, A., Quilez, J. and Regot, J.*: **Sunlight Energy Graphic and Analytic Control in 3D Modelling**; (7) *Kokosalakis, Jen*: **Learning to Learn Through Computing: Sensitising Surveys and Empowering Urban Stakeholder's Input to Policy**; (8) *Loy, Hollis A.*: **Foundation for a Thorough CAAD Education**; (9) *Martens, Bob and Turk, Ziga*: **Working Experiences with a Cumulative Index on CAD: "CUMINCAD"**; (10) *Martens, Bob*: **Education in Computer Aided Architectural Design in Europe**; (11) *Maver, T. and Petric, J.*: **The Future Will Be Just Like the Past: Only More So: A Tribute to the Late John Lansdown**; (12) *Murison, Alison*: **Less is More - Enhancing CAD Instruction by Enabling Student Centred Learning, a Case Study for Learning 2000**; (13) *Sanchez, S., Zulueta A., and Barrallo J.*: **Bilbao: The Revitalisation of a City**; (14) *Shih, Naai-Jung and Huang, Yen-Shih*: **An Analysis and Simulation of Curtain Wall Reflection Glare**; (15) *Tournay, Bruno*: **The Software Beats the Hardware**; (16) *Tweed, Christopher*: **Prescribing Designs**; (17) *Voigt, A., Walchhofer, H.P. and Linzer, H.*: **The Historico-cultural Past as Spatial-related Cognition Archives: Computer-assisted Methods in the History of Urban Development, Archeology and History of Art**; (18) *Wingham, Ivana*: **Digital Space, Social Technology and Virtual Force as Determinants of Design in the 21st Century**;

Cluster 5: system model design form idea dynam analyt comput reason behaviour

(1) *Coates, Paul and Schmid, Claudia*: **Agent Based Modelling**; (2) *Jozen, T., Wang, L. and Sasada, T.*: **Sketch VRML - 3D Modeling of Conception**; (3) *Lentz, Uffe*: **Integrated Design with Form and Topology Optimizing**; (4) *McFadzean, Jeanette*: **Computational Sketch Analyser (CSA): Extending the Boundaries of Knowledge in CAAD**; (5) *Mishima, Yoshitaka and Szalapaj, Peter*: **ADMIRE: an Architectural Design Multimedia Interaction Resource for Education**; (6) *Montagu, Arturo, Rodriguez Barros, Diana and Chernobilsky, Lilia B.*: **Design, Qualitative Analysis and Digital Media: An Experimental Pedagogic Approach to the Cultural Evaluation and Integration of Media** ; (7) *Naticchia, Berardo*: **Physical Knowledge in Patterns: Bayesian Network Models for Preliminary Design**; (8) *Reffat, Rabee M. and Gero, John S.*: **Situatedness: A New Dimension for Learning Systems in Design**; (9) *Wang, L., Jozen, T. and Sasada, T.*: **Construction of a Support System for Environmental Design**; (10) *Yakeley, Megan*: **Simultaneous Translation in Design: The Role of Computer Programming in Architectural Education**;

Figure 9. Content of some clusters generated by Crossbow.

3.4 Clustering by a human

Andy Brown, Michael Knight and Philip Berridge (1999), the editors of the 1999 eCAADe proceedings organised the proceedings into the following clusters:

Historical Perspectives
The Creative Process
The Education Process
Communicating Information and Ideas
Collaborative Learning and Design
Virtual Environments
Virtual Design Studio
The Rendered Image
The Intelligent Machine 1: AI
The Intelligent Machine 2: Generative and Genetic
Planning and Urban Design
Architectural Technology
The CAAD Education Web

Planning and Urban Design

Architects in the Information Society: the role of New Technologies
Rossella Corrao, Giovanni Fulantelli

The Historico-cultural Past as Spatial-related Cognition Archives: Computer-assisted Methods in the History of Urban Development, Archeology and History of Art
Andreas Voigt, Hans Peter Walchhofer and Helena Linzer

Autonomous Spatial Redistribution for Cities
Jon Moorhouse and Gary Brown

Three dimensional computer models in Development Planning
Wolfgang Dokonal

Bilbao: The revitalisation of a City
Sanchez S., Zulueta A., Barrallo J.

SOL: Spatial and historical web-based interface for On-line architectural documentation of Kraków's Rynek Główny
Jean-Yves Blaise and Iwona Dudek

IT in Urban Regeneration Projects
Steen Holmgren and Bjarne Rüdiger,

Figure 10. Clusters as made by a group of human editors (left); sample cluster made by a human (right). The similarity with the machine-generated clusters of the previous sections is almost non-existent.

4. CONCLUSIONS

It certainly is possible to find some sense in the clusters created by the machine and these clusters can be used to suggest a user of CUMINCAD to look at a few other papers, in addition to the ones she found. In this way she would examine more works and stay on the site longer. The functionality will be made part of CUMINCAD during the first half of 2001.

However, the machine made clusters are much different to the clusters created by the humans. It seems hard for a machine without the background deep knowledge to cluster a topic on its own and, in this way autonomously define what the topics of CAAD are. Indeed the presented algorithms have numerous parameters by which the results can be tweaked, but, after extensive tweaking by humans, how "machine generated" would the results then be?

This confirms our belief that the way we understand the topics of CAAD, and into which this or that paper belongs to, is subjective and based on one's the current interests and perspectives. What defines a scientific community is, that its members, to a large extent, also share a similar deep understanding of the topic.

A machine can learn this shared perspective in several ways. One is through parameter tweaking of the presented algorithms – initial tests show that best results are achieved by manually expanding the list of stop words.

Through this approach, however, the machine would only learn a view of the tweaker.

4.1 Future work

Our future work will be dedicated to learning about the shared perspective in the same way as the humans did – through interaction with the members of the scientific community. For example, the machine can learn from the humans that use a database by observing what papers a single user in a certain time frame looks at - it is very likely that these papers belong to the same topic. Similarly, man made clusters e.g. from conference proceedings, could be used as a learning example through which the machine could learn to classify in the same way, as that conference organisers did. In this way, the machine can learn the communities' view and pass on this view onto the new members of the community. By observing the users, browsing through the classes, the machine could establish which classifications are correct and which not.

While the learning from raw data failed in our case, we believe that in collaborative Web based applications, the real potential of machine learning is in the learning from the users. The implementation is rather simple - in the next iterations of the clustering we plan to add one extra field to the extracted data set – the name of the reader who demonstrated interest in a particular record. This data is readily available from the server log files.

4.2 Future research topics

The presented methods are not relevant only in the context of scientific papers but with any other text databases related to engineering, such as building codes, historical project data, best practice guides etc. Same techniques, as presented, can also be used to ease the navigation and increase the relevance of the search results.

Text, however, is not the format in which architects and engineers would encode most information. We are used to work with drawings and product data. Computer scientists developed numerous algorithms for text learning, clustering and classification. All these algorithms work with text. However, as we have shown in Section 2 only initial steps in the algorithms parse text. Afterward only numbers are crunched. To take advantage of such technologies, CAD community needs to develop algorithms for selecting features, extracting features and vectorisation of drawings and product data.

5. REFERENCES

Brin, S. and Lawrence, P. (1998). The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Science Department, Stanford University, Stanford, CA, USA, http://www7.scu.edu.au/programme/fullpapers/1921/com1921.htm (modified 23 March 1998).

Brown, A., Knight, M. and Berridge, P. (Eds.) (1999) Architectural Computing from Turing to 2000 [Conference Proceedings], eCAADe Conference Proceedings / ISBN 0-9523687-5-7 / Liverpool (UK) 15-17 September 1999, 773 p.

Christiansson, P. (1998). Knowledge Nodes, in I. Smith (ed) Artificial Intelligence in Structural Engineering, Springer, Berlin, 197-213.

Hovestadt, Ludger (1993) A4 Digital Building: Extensive Computer Support for Building Design, Construction, and Management, CAAD Futures '93 [Conference Proceedings / ISBN 0-444-89922-7] (Pittsburgh / USA), pp. 405-421.

Maher, M.L. and Simoff, S. (1998). Knowledge discovery from multimedia case libraries, in I. Smith (ed) Artificial Intelligence in Structural Engineering, Springer, Berlin, 197-213.

Martens, B. and Turk, Z. (1999) Working Experiences with a Cumulative Index on CAD: "CUMINCAD", Architectural Computing from Turing to 2000, eCAADe Conference Proceedings / ISBN 0-9523687-5-7, Liverpool, (UK) 15-17 September 1999, pp. 327-333

McCallum, A.K. (1996). "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering.", http://www.cs.cmu.edu/~mccallum/bow, 1996.

Mladenic, D. (1999). Text-learning and related intelligent agents, IEEE Expert special issue on Applications of Intelligent Information Retrieval, July-August 1999

Salton, G. (1989). Automatic Text Processing. Addison-Wesley, Reading MA, 1989.

Scott, S. (1998). Feature Engineering for a Symbolic Approach to Text Classification, Ms. Thesis, University of Ottawa Computer Science, 1998.

Seni, D. A. and Hodges, W. (1996). Common grounds and relationships between information technology and philosophy. Association for Information Sciences, America's Conference 96. 2: 398-400.

Turk, Ž. (1999). WODA: A Slim Web Oriented Database, Heterogeneous and Internet Databases, City University of Hong Kong Press, Hong Kong, 1999, ISBN: 962-937-046-8, pg. 95-109.

Tweed, Christopher (1999) Prescribing Designs, Architectural Computing from Turing to 2000, [eCAADe Conference Proceedings / ISBN 0-9523687-5-7] Liverpool (UK) 15-17 September 1999, pp. 51-57.

van Rijsbergen, C.J (1979). Information Retrieval, 2nd ed.. London, Butterworths, 1979.

Willett, P. (1988). Recent Trends in hierarchic document clustering: A critical review. Information Processing and Management, Vol. 24, No. 5, 1988.

Zamir, O., Etzioni, O. (1998). Web Document Clustering: A Feasibility Demonstration (SIGIR 1998).

Zamir,O and Etizioni, O (1999). Grouper: A Dynamic Clustering Interface to Web Search Results, Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195-2350, USA, http://www.cs.washington.edu/research/clustering (modified 09 June 1999).

Computer-based TRIZ - Systematic Innovation Methods for Architecture

Darrell L Mann[1] and Conall Conall Ó Catháin[2]
University of Bath[1], Queen's University Belfast[2]

Key words: Architectural Design, Concept Design, Creativity, Innovation, Knowledge-management, complexity-management, TRIZ

Abstract: The Russian Theory of Inventive Problem Solving, TRIZ, is the most comprehensive systematic innovation and creativity methodology available. Essentially the method consists of restating a specific design task in a more general way and then selecting generic solutions from databases of patents and solutions from a wide range of technologies. The development of computer databases greatly facilitates this task. Since the arrival of TRIZ in the West at the end of the Cold War, it has begun to be used with great success across a wide variety of different industries. Application of the method to the field of architecture has so far been very limited. The paper outlines how TRIZ methods may be applied to a number of architectural problems.

1. INTRODUCTION

Most architects recognise the importance of innovation. Most also equate the idea of innovation to high risk. Whether related to new or improved constructions, construction components, processes or services, the innovation process – if indeed it can be called a process at all – is viewed as a nebulous, unpredictable activity with little or no degree of certainty in terms of either output quality, cost or time.

The Russian initiated Theory of Inventive Problem Solving, TRIZ, (1, 2) looks set to do much to change this picture. TRIZ offers architects looking for inventive solutions the prospect of a manageable, predictable, systematic innovation capability. The method has been built upon over 1500 person years worth of research into the inventive process and the study of nearly 3

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 561–575.
© 2001 *Kluwer Academic Publishers.*

million of the world's most successful patented inventions. The research has established that:

1. Problems and solutions are repeated across industries and technologies
2. Patterns of technical evolution are repeated across industries and technologies
3. The most powerful innovations use effects and solutions from outside the field where they were developed
4. The most powerful innovations are the ones that eliminate rather than accept compromises

Since its emergence in the West in the late 1980s, the method has begun to be successfully deployed both as a manual method and a computer-based tool by a number of companies in the US and, progressively, Western Europe and Japan. Although initially conceived as a method for engineers, TRIZ has latterly been seen to be successfully applied to a much wider variety of problem types – including non-technical, business type problems. Exposure to TRIZ across the field of architecture has thus far been extremely limited.

This paper discusses how an evolved version of TRIZ, with supporting software tools, is being used to solve previously intractable problems and, using the identified technology evolution patterns, to define the future development potential and paradigm shift potential of a range of building systems and sub-systems. The paper includes examples of TRIZ-derived solutions to a range of architecture-based problems in order to demonstrate how the method may be expected to have as profound an impact on innovation in architecture as it has had in other sectors.

The paper begins with a brief overview of TRIZ and the problem definition and solving tools contained within it.

2. TRIZ – AN INTRODUCTION

The core findings of TRIZ research on the global patent database (Reference 1, 2) are that the world currently contains a very small number (40) of Inventive Principles and that technology evolution trends are predictable.

TRIZ provides means for problem solvers to access the good solutions obtained by the world's finest inventive minds. The basic process by which this occurs is illustrated in Figure 1. Essentially, TRIZ researchers have encapsulated the principles of good inventive practice and set them into a generic problem-solving framework. The task of problem definers and problem solvers using the large majority of the TRIZ tools thus becomes one in which they have to map their specific problems to and select solutions from this generic framework.

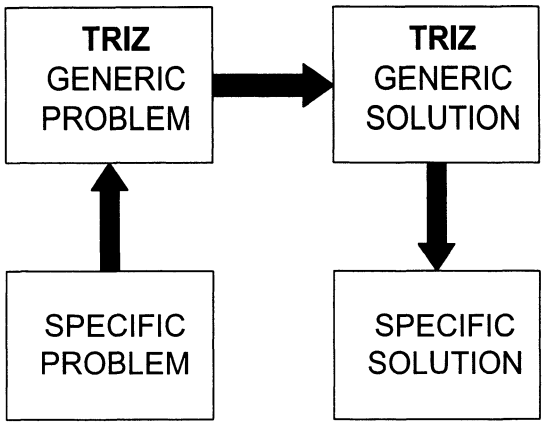


Figure 1 The Basic TRIZ Problem Solving Process

By using the global patent database as the foundation for the method, TRIZ effectively strips away all of the boundaries which exist between different industry sectors. The generic problem solving framework thus allows engineers and scientists working in any one field to access the good practices of everyone working in not just their own, but every other field of science and engineering.

Readers interested in obtaining a flavour of the breadth of applicability should check out some of the several hundred articles in TRIZ Journal (3).

3. THE FOUR PILLARS OF TRIZ

1500 person years of research have produced a lot of significant innovation tools and methods. TRIZ allows users to deploy each of these tools in either an individual or systematically sequenced manner. Experience using the method in non-Russian work environments has suggested that users often struggle with the various tools and techniques because it is difficult to set TRIZ in the context of 'traditional' problem solving strategies. With this in mind, the description offered here re-casts TRIZ in order to strengthen awareness and understanding of the four main paradigm shifts which distinguish the methodology from others.

The four paradigm shifts – Contradiction, Ideality, Functionality, and Use Of Resources (Figure 2) are discussed below in the context of the central role they each play in creating a deployable and effective problem definition and problem solution methodology.

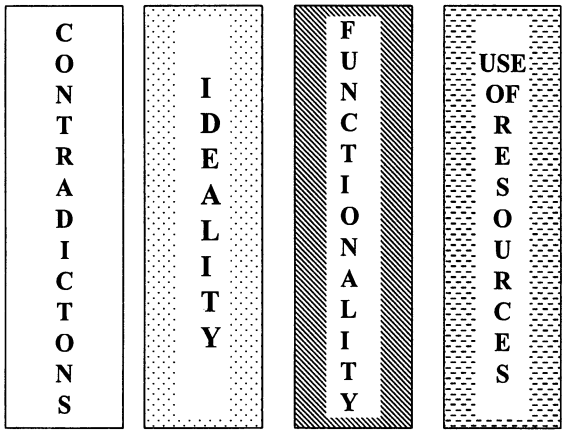


Figure 2. The Four Pillars of TRIZ

3.1 Contradictions

Although often the first of the tools seen by newcomers to TRIZ, Contradictions is probably the tool which is deployed least well. At least part of the reason for this is that the main underlying principle of the Contradictions philosophy – that of seeking to identify and eliminate contradictions – is almost the complete opposite of traditional problem solving strategies, in which the emphasis is very firmly placed on the importance of achieving 'optimum' compromises between conflicting problem parameters. Traditionally, there is a strong tendency to think of the design process as an amorphous bag filled with an incompressible fluid made from the different design parameters, in which, as the designer tries to squash the bag to improve one parameter, it bulges out somewhere else as a different parameter gets worse.

The keen emphasis on 'trade-off' solutions in traditional problem solving practice often means that designers are rarely explicitly aware that conflicts exist. The first major part of the paradigm shift that takes place in the Contradictions part of TRIZ is the need for problem solvers actively to seek out the conflicts and contradictions inherent in all systems. The second part then involves using the TRIZ methodology to try and 'eliminate' (4, 5) those contradictions rather than to accept them. Or, in terms of the incompressible-fluid filled bag analogy, to attach a valve of some kind that allows the amount of fluid in the bag to be altered.

TRIZ contains a number of 'contradiction elimination' tools – primarily the Contradiction Matrix (2) – which encapsulate how others have successfully solved similar problems. TRIZ currently identifies 40 Inventive Principles which might apply in any given contradiction situation. The

Contradiction Matrix allows problem solvers to narrow down that list of 40 to three or four Principles which might apply to an individual contradiction type. The discovery of the 40 Principles does not preclude the existence of a 41st or indeed many more, merely that today inventors have used just 40. Recent research (6) has highlighted the fact that the same principles are also applicable to non-technical problems.

3.2 Ideality

TRIZ founder, Genrikh Altshuller identified an overall trend of evolution for technical systems which states that they will always evolve towards increasing 'ideality.' and that this evolution process takes place through a series of evolutionary S-curve characteristics (1, 7). A key finding of TRIZ is that the steps denoting a shift from one S-curve to the next are predictable. This finding may be expected to play a significant role in helping organisations to predict how and when evolution steps are possible. This is an undoubtedly useful capability when seen relative to the manner in which organisations have traditionally viewed the innovation process.

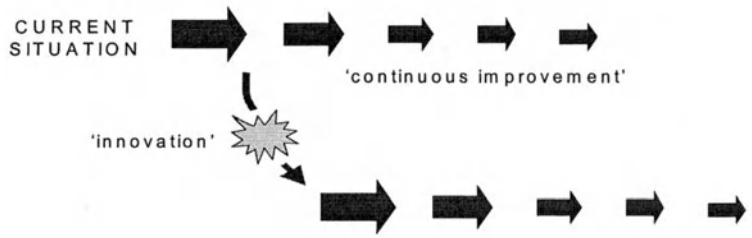


Figure 3. : Traditional System Improvement and Evolution Strategy

Figure 3 illustrates a typical traditional system evolution path. The improvement of systems using this method of operating is very much rooted in the existence of the current situation, and all improvement initiatives use the current design as their foundation. The large-scale innovations then usually appear through what is usually perceived as a highly random process. According to TRIZ and other research, there is a very high likelihood that these major innovations will come from outside the existing industry. In fact, the likelihood is close to 100% (9).

The essential paradigm shift between this approach and the TRIZ approach is that while traditionally, problem solvers start from the knowns of today, the concept of Ideality, demands a strategy in which the problem solver is first asked to eliminate the constraints of today's solution, to then envisage the 'ideal final result' situation – in TRIZ terms where the function

is performed without any resource, cost or harm – and to use that as the basis from which a realisable solution is derived. The problem solver may thus be seen to be working back from the 'ideal' to something which is then physically capable of being engineered. This strategy is illustrated in Figure 4. Several examples of this strategy in operation can be found in (9,10).

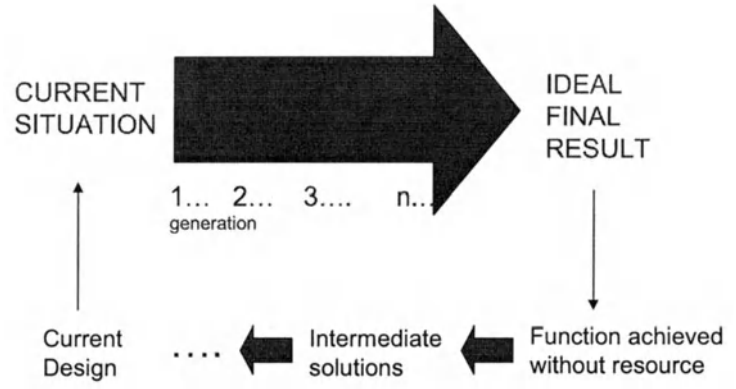


Figure 4. Proposed 'Ideality-Based' Improvement and Evolution Strategy

As well as offering a successful evolution strategy and real problem solutions, it may also be noted that the method also provides a considerable amount of valuable long-term strategy definition data.

3.3 Functionality

Although the functionality aspects of TRIZ owe a significant debt to the pioneering work on Value Engineering by Miles (11), the method of defining and using functionality data is markedly different. It is sufficient at the very least to merit discussion as a distinct paradigm shift in thinking relative to traditional occidental thought processes. Three aspects are worthy of particular note:-

1. The idea that a system possesses a Main Useful Function (MUF) and that *any system component which does not contribute towards the achievement of this function is ultimately harmful.* In a heat exchanger, for example, the MUF is to transfer heat to the working medium; everything else in the system is there solely because we don't yet know how to achieve the MUF without the support of the ancillary components. (Systems may of course perform several additional useful functions according to the requirements of the customer.)
2. In traditional function mapping, the emphasis is very much on the establishment of positive functional relationships between components.

TRIZ places considerable emphasis on plotting both the positive *and the negative relationships* contained in a system, and, more importantly, on using the *function analysis as a means of identifying the contradictions* in a system.

3. Functionality is the common thread by which it becomes possible to share knowledge between widely differing industries. A motor car is a specific solution to the generic function 'move people', just as a washing powder is a specific solution to the generic function 'remove solid object'. By classifying and arranging knowledge by function, it becomes possible for manufacturers of washing powder to examine how other industries have achieved the same basic 'remove solid object' function. '*Solutions change, functions stay the same*' is a message which forms a central thread in the TRIZ methodology÷

The emphasis TRIZ places on functionality demands that problem solvers adopt a much more flexible approach to the way in which they look for solutions to their problems. The age of the specialist is coming to an end; it is no longer sufficient for mechanical engineers to only look for mechanical solutions to their problems when someone from, say, the chemical sector may already have discovered a better way of achieving the function being sought – Figure 5.

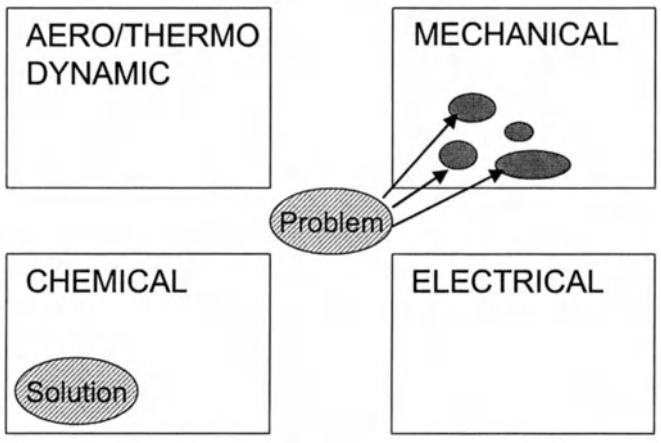


Figure 5. Solution Spaces

Functionally classified knowledge databases are now becoming commercially available. Probably the most comprehensive – currently containing around 6000 effects and examples – comes from Invention Machine (12). One of the following examples will provide a demonstration of the use of this kind of knowledge base in helping to solve architecture problems.

3.4 Use Of Resources

The last of the four main paradigm shifts contained within TRIZ is the simplest, and relates to the unprecedented emphasis placed on the maximisation of use of everything contained within a system. In TRIZ terms, a resource is *anything in the system which is not being used.* TRIZ demands an aggressive and seemingly relentless pursuit of things in (and around) a system which are not being used to their maximum potential. Discovery of such resources then reveals opportunities through which the design of a system may be improved.

In addition to this relentless pursuit of resources, TRIZ demands that the search for resources also take due account of negative as well as the traditionally positive resources in a system. This is done because experience has demonstrated that the discovery of a negative resource coupled with application of the 'Blessing In Disguise' Inventive Principle can often lead to significant design improvements.

For example, Russian engineers often think of resonance as a resource. This is in direct contradiction to most Western practice, where resonance is commonly viewed as something to be avoided at all costs. TRIZ says that somewhere, somehow, resonance in a system can be used to beneficial effect. In effect, resonance is a potent force lever capable of amplifying small inputs into large outputs. Resonance is currently being used to generate beneficial effects in a number of new product developments from vacuum cleaners (resonating brush fibres to enhance extraction of dust particles), paint stripping systems on ships (firing a pulsed jet of water – existing resource! – at the local resonant frequency of the hull), and in helping to empty trucks carrying powder-based substances more quickly.

The following case study examples serve to demonstrate how the different TRIZ tools and techniques can be brought together to give problem solvers new perspectives on the way they think about the systems they create:

4. EXAMPLE 1 – 'WINDOWS'

Access to enable window cleaning to take place is a functional requirement seemingly forgotten about in too many building designs. Window cleaning is traditionally done by a manual soap-and-water operation. TRIZ encourages problem solvers to think about the functionality to be delivered – in this case –'clean a surface' and to then use that as a

means of identifying if there may be other means of achieving the same function. A functionally arranged knowledge base allows us to tap into the good and potentially relevant solutions to the same functions we require to deliver. In the case of 'clean a surface', the current knowledge bases do not map directly onto our requirement, but providing we can make the jump between our problem and the generic knowledge framework contained in the database, we have a means of accessing the solutions of others – Figure 6.

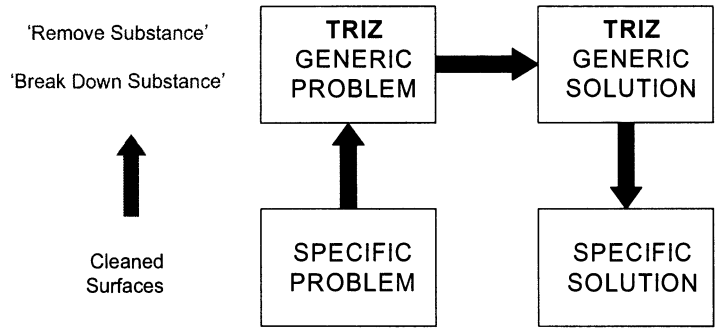


Figure 6. Abstraction of 'Clean Surface' onto Knowledge Base.

In this case, the knowledge base will then show us that many other problem solvers have already successfully achieved solutions to the functional requirement – Figure 7.

While several of these solutions may offer an improved method over current strategies, we will choose instead to extend the overall approach to think about the ideal final result cleaned surface situation – a surface that cleans itself. The word 'self' is very important in a TRIZ context and 'things that achieve functions for themselves' offer great benefits to customers. The functional database of solutions to the 'self-cleaning' requirement is somewhat smaller than the above, but nevertheless contains reference to the Lotus Effect. The Lotus Effect – named after the plant – points us towards the very effective manner in which the leaves of the plant clean themselves.

Having found a potential solution to the functional requirement, the problem solver is then required to translate the generic solution into a specific solution. In the case of 'self-cleaning glass' using the Lotus Effect, that process is currently the subject of an ongoing patent application. In the meantime, a similar desire for 'self-cleaning facades has already led to the development and launch of a 'self-cleaning' façade paint (13).

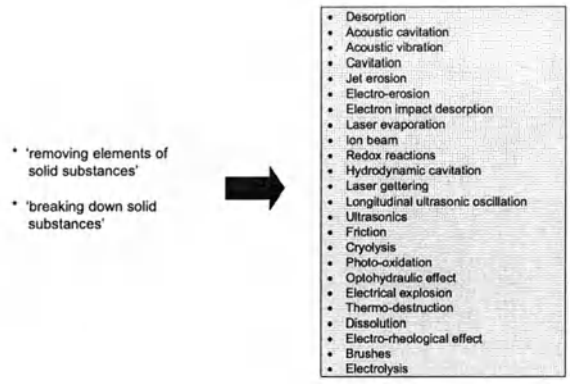


Figure 7. Many Ways of Achieving the Function 'Clean Surface'

5. EXAMPLE 2 – 'RAMPS'

Access ramps are a blight on many a public building, being both aesthetically unpleasing, and a source of inconvenience for the vast majority of users including those for whom they are provided. Ramps represent the lowest common denominator design compromise. In TRIZ terms, every time an architect sees compromises, the method is trying to suggest better alternative solutions.

The TRIZ Contradiction Matrix offers systematic means of tapping into the good solutions of inventors who have successfully fought rather than accepted the compromises. In the case of 'ramps', the design contradiction centres on a desire to get up and down steps of a variety of heights with the minimum level of inconvenience. The 'compromise-free' solution then is one that allows us to deliver the function (get up and down steps) **and** not have any inconvenience to any user. In the generic terms of the Matrix, this contradiction corresponds to one of length versus ease of operation. The Matrix then suggests that the best inventive solutions to this generic conflict have involved either transferring the delivery of the function to something else, or getting the thing that wants the function to achieve it 'by itself'.

These generic solution triggers are often very useful ways of changing our perspective on a problem. In this case, they are helping the problem owner to stop thinking about a 'ramp', and to start thinking about how the wheelchair user (the person that wants the function delivered) can deliver the function. This shift in thinking might then point us towards use of a knowledge base to help us to see if someone else has successfully made the same sort of shift in thinking. In this case, we might examine the global patent database in search of inventions which have delivered the function 'raise (or lower) human (in a wheelchair, up or down a step or series of steps)'. From a list of over 70 potentially relevant inventions, we find one – US patent US5701965: 'Human Transporter' (Figure 8) – which has recently entered the market in the United States.

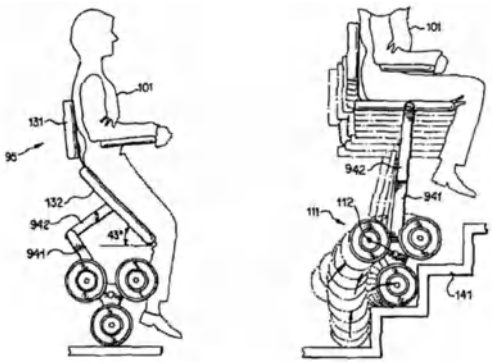


Figure 8. US Patent 5701965, Human Transporter

Of course, this solution is but one of a wide range of possibilities which also include 'wheelchairs' that successfully break out of the psychological inertia associated with the use of 'wheels' by employing 'legs'. And, although it performs many functions in addition to the 'raise human' one relevant to the ramp issue – for example its (low cost) gyroscopic control system allows the occupant to stably rise up on two wheels to achieve eye-to-eye contact with a standing person – it comes with compromises of its own. In the fullness of time, TRIZ techniques will facilitate the elimination of these compromises. In the meantime, if it serves only to help architects and regulation authorities to break out of the 'ramp' paradigm and into one that encourages solution closer to source at the wheelchair level, it has served a useful purpose.

6. EXAMPLE 3 – 'LIFETIME HOMES'

Staying with the theme of raising and lowering humans, but now extending the function to a desire to move between storeys by means other than the specific solution of a flight of stairs, we comment on current suggestions that the designs of new domestic dwellings should include special provision in the construction of the first floor to facilitate the future installation of a lift in order to accommodate aging occupants that become unable to use a staircase.

Again, here we see a case of design guidance specifying a design solution rather than focusing on the delivery of the desired function. A 'lift' is but one of a number of ways of moving between levels within a building; just as 'a staircase' is another. The point of switching from a solution-based to a function-based way of looking at the world is that while solutions can and often do change, functions stay the same. In other words, humans will always want to go up and down between different levels, but likely as not also, they may well wish to achieve that function by means other than those which blinker our thinking today.

Even worse in this case is the potential requirement that buildings will have to have redundant structure permitting the future **possibility** of having a lift installed. If we also allow ourselves to be bound by the psychological inertia that tells us that lifts travel vertically up and down, we have imposed some massively limiting design constraints on architects and on our society. Most 'lifts' do travel vertically up and down; but that doesn't mean all have to. Nor does it mean that the means of delivering the function has to either.

Again, the use of a TRIZ-based functional knowledge search of the US patent database, for example, reveals over 70 patents related to powered means of transition between floors in buildings.

The Stannah-style stairlift is a relatively crude alternative solution for delivering the function. Crude as it is (lots of contradictions for TRIZ to help eliminate!), the design concept has at least made use of an existing resource. A multi-storey building with a staircase has a ready made means of accessing one floor from another – so why force builders to include another? One of the problems with the Stannah design concept is that it demands attachment to a structural wall. While this may not be too major a constraint, in TRIZ terms, it is one that 'ideally' we would not impose on ourselves. Thus US patent 5105914 (Figure 9) delivers the same functionality as the Stannah, but does not require attachment to anything other than an existing resource called the staircase, nor does it require installation on a footprint bigger than the staircase.

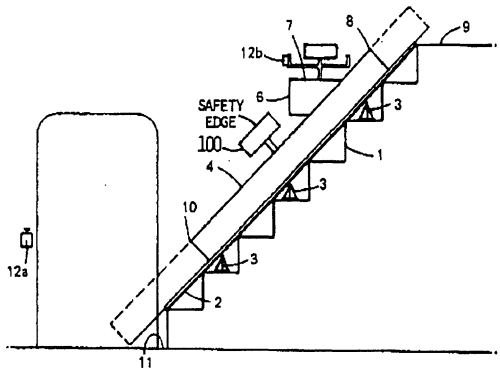


Figure 9. US Patent 5105914, Stairlift

7. EXAMPLE 4 – 'STAYING WARM'

We end with another short case study aimed at reinforcing the importance of recording the functions and attributes required from a given structure, and then using software knowledge bases to identify possible solutions to each functional requirement. The example here examines the functional requirement of a building façade to provide insulation. Again, 'insulation' is not a problem unique to the field of architecture; it is a functional requirement found in most if not all other scientific and engineering disciplines.

For the first time, TRIZ-based software gives problem solvers ready access to the 'good' functional solutions discovered and exploited in these other disciplines. Potentially, several of those solutions offer significant benefits over the ones 'traditionally' used within any given individual field.

In the case of 'insulation' – or, in terms of the generic functional classification used in the knowledge-bases, 'stabilise thermal parameters' – there are close to 200 scientific effects capable of delivering the function (Figure 10). Many of the list appear to offer significant opportunities for architectural innovation. To access them, however, will demand that architects adopt new ways of looking at the world outside their discipline.

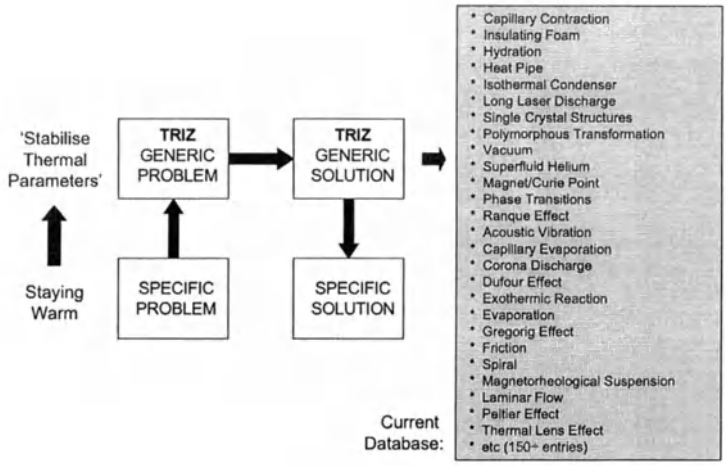


Figure 10. How Other Fields Achieve the Function 'Stay Warm

8. SUMMARY

1. The systematic innovation and creativity methodology, TRIZ, is formed around four main pillars, contradictions, ideality, functionality and use of resources.
2. TRIZ is a comprehensive systematic innovation and creativity method with a massive pedigree of published output. It offers a generic problem-solving framework into which other tools have already begun to be successfully integrated.
3. TRIZ has not previously been applied in the field of architecture, to the authors' knowledge. Preliminary evidence from the case studies presented here suggests that the method offers much potential for systemising the innovation process in architecture.
4. TRIZ-based knowledge bases enable architects to readily identify and explore the good solutions of others in other fields.
5. Such knowledge bases would benefit from expansion to incorporate the 'good solutions' known about by architects that have not been discovered in other disciplines.

9. REFERENCES

Altshuller, G., 1984, *Creativity As An Exact Science*, Gordon & Breach.

Salamatov, Y., 1999, *TRIZ: The Right Solution At The Right Time*, Insytec BV, The Netherlands

Mann, D.L., 2000, "Contradiction Chains", *TRIZ Journal*, January 2000.

Mann, D.L., 1999, "*Design Without Compromise: A New Perspective on Fluid Power System Component Design*", *PTMC International Workshop*, PTMC99, Bath, September 1999.
Mann, D.L., E. Domb, 1999, "40 Inventive Management Principles With Examples", *TRIZ Journal*, August 1999.
Mann, D.L., 1999, "Using S-Curves and Trends of Evolution in R&D Strategy Planning", *TRIZ Journal*, July 1999.
Utterback, J, 1996, *Mastering The Dynamics Of Innovation*, Harvard Press.
Mann, D.L., 2000, "Strength versus Weight Contradictions in Bridge Design", *TRIZ Journal*, March 2000.
Mann, D.L., 2000, "Design Without Compromise, Design For Life", *International Fluid Power Exposition*, IFPE 2000, Chicago.
Miles, L. D., 1961, *Techniques of Value Analysis and Engineering*, McGraw-Hill Book Company, New York NY.
www.invention-machine.com
www.botanik.uni-bonn.de/system/bionics.htm
www.triz-journal.com

FormWriter

A Little Programming Language for Generating Three-Dimensional Form Algorithmically

Mark D. Gross
Design Machine Group, Department of Architecture, University of Washington

Key words: Programming Language, Geometry, Form Generation

Abstract: FormWriter is a simple and powerful programming language for generating three-dimensional geometry, intended for architectural designers with little programming experience to be able to generate three dimensional forms algorithmically without writing complex code. FormWriter's main features include a unified coding and graphics environment providing immediate feedback and a "flying turtle" - a means of generating three dimensional data through differential geometry.

1. ALGORITHMIC FORM GENERATION

1.1 Introduction

Powerful three-dimensional modelling software has made it easy for architects to shape and sculpt three dimensional material and space using direct manipulation operations. Architects find that computer aided design modellers can help them generate radically new forms for buildings. Gehry's forms in the Guggenheim museum in Bilbao and at the Experience Music Project in Seattle, Greg Lynn's proposals for blob architecture (Lynn, 1998); Reiser+Umemoto's IFCCA Competition for the Design of Cities project in New York (Benjamin and Libeskind, 1998)—all show radical departures from conventional forms enabled by computer aided design software. (Kolarevic, 2000) provides an overview of these developments, some of which are mentioned in (Perrella, 1998; Toy, 1999).

With these novel forms has come a renewed interest in generating form algorithmically, that is, writing computer programs whose execution results in three dimensional geometry, and using this computational medium to

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 577–588.
© 2001 *Kluwer Academic Publishers.*

explore architectural form. Generative systems are a well-established theme in computer-aided architectural design, including the approaches of shape grammar, genetic algorithms, and parametric variation. However, algorithmic form generation has largely remained the province of academic investigation, perhaps because of the level of technical expertise it has required.

Visual arts and music have a long history of exploring algorithmic form generation. Computer music, from the early days of Illiac, has been largely engaged in the development and deployment of sound producing algorithms. Similarly algorithmic generation of images has been a central theme in visual art. Although algorithmic form generation has certainly had a presence in computer-aided architectural design, it is fair to say that other interests have taken a more prominent position.

Investigating new forms for buildings, freed from the constraints of conventional building materials and techniques, architects are beginning to explore how algorithms can be used to generate complex three-dimensional curves, arrangements, and folding of space and material. Algorithmic generation can produce significantly different and more complex forms than conventional CAD.

Despite the power of industrial strength 3D modellers such as CATIA and Maya, many things are difficult to accomplish, and some just can't be done. It remains difficult to use most CAD modellers (Revit notwithstanding) to generate three dimensional forms through parameterisations and more sophisticated algorithms. Algorithmic form generation is familiar in the engineering design disciplines: civil and aeronautical engineering and naval architecture, where three dimensional shapes are more strictly dictated by functional requirements expressed as mathematical equations.

If one wants to generate three dimensional forms algorithmically, one must decide between two main alternatives: On the one hand macro facilities and scripting languages within CAD modellers are relatively easy to learn, but they inherently limit the programs one can write (and hence the forms one can generate). On the other hand, full-fledged programming languages such as C and Java are powerful but they require more effort to learn, and generating 3D geometry also requires attention to many language features that have no direct bearing on form.

1.2 Support for algorithmic form generation

Table 1 distinguishes five levels of support for algorithmic form generation provided by CAD modellers. The simplest modellers provide no support whatsoever for algorithmic generation: models must be constructed directly using the geometric primitives and operations provided on the CAD modeller's menus. Most CAD modelling programs offer macro facilities or

Table 1. Five levels of support for algorithimic generation

1	None	the designer may use only the geometric primitives and operations built in to the modeller.
2	Macros	frequently used sequences of operations can be recorded and replayed, in some cases allowing parameters to be supplied at replay time.
3	Scripting languages	more control over the modeller than recorded macros but which fall short of a full-fledged programming language.
4	Embedded programming language	e.g., AutoCAD's AutoLisp or ArchiCAD's GDL. Access to the modeller's library via a language within the modeller.
5	External programming language	programs typically written in C or Java communicate with and control the modeller. Bentley Microsystem/J

scripting languages. Although a macro facility serves simple tasks well (such as repetitive window patterns or stairs), it is difficult to program more complex operations using only macros. Scripting languages, which have gained wide acceptance in other domains (witness JavaScript and Flash), provide considerably more power than macros but coding more sophisticated tasks becomes quite complex, requiring a specialist programmer. An embedded programming language, like a scripting language, enables the programmer to control and command the modeller from an environment within the CAD program, and allows more powerful constructs than the typical scripting language. Many CAD programs now include an embedded language, and advanced users of these CAD programs enthusiastically endorsetheir modeler's scripting or embedded language. AutoLisp is arguably the best known example. Although the underlying Lisp language is extremely elegant and powerful, Autodesk's implementation was a weak one and the programming environment for developing AutoLisp routines is woefully inadequate by modern standards. Another example of an embedded programming language is ArchiCAD's GDL, which provides access to the modeller's functionality through a BASIC-like language. Although it provides this functionality, the choice of a BASIC programming style limits the language and renders it inelegant: Language design makes an enormous difference. GDL does enable the construction of parametric objects. A fully fledged programming language such as C or Java can be used to write complex form-generating algorithms but it requires more expertise than most designers are willing to commit to acquiring. For a

recent example in visual art see (Berzowska, 1998); an architectural example is Terzides's Java based experiments in morphing (Terzidis, 1999).

Some designers have resorted to using software such as Mathematica or MathCAD to generate three-dimensional surfaces. Mathematical software is good at describing three dimensional surfaces by parameterised functions, but provides limited support (at best) for the basic features that make programming a powerful medium of expression: for example, conditional expressions, iteration, data structures such as sequences and strings. What designers need is a simple way to explore and generate forms algorithmically, without the complexities of a professional programming language.

1.3 FormWriter

FormWriter is an easy-to-use programming language designed especially to allow architects and architecture students to explore algorithmic form generation. The idea of FormWriter is to eliminate complexity without sacrificing the power of programming. FormWriter offers a simple syntax, a unified development environment, and easy access to three-dimensional libraries.

The FormWriter program described below follows in a tradition of novice and domain-oriented programming languages (du Boulay, 1981; Bentley, 1986; Tweed, 1986). The time may now be ripe for a new, architectural, generation of these 'little languages'. Powerful 3D graphics are now available on the desktop and architects and students are better prepared to write code. Like John Maeda's "Design By Numbers" little language for graphic desingers, (Maeda, 1999) the design of FormWriter derives from experience with an earlier generation of novice programming languages at the MIT Logo Project (Papert, 1980).

With only a few lines of code a designer can generate three dimensional graphics immediately, and within minutes can explore parameterised and conditional construction to generate complex combinations of forms. The graphics environment is integrated with the code editor and programming environment, allowing a designer to explore forms fluidly without the distractions of a code-compile-load-execute cycle.

FormWriter is a domain-specific language for novice programmers; nevertheless it a full-fledged language with constructs for passing arguments and returning values, conditional execution, iteration and recursion.

2. FORMWRITER

Figure 1 shows the FormWriter working environment: at left a window into a 3D space with browsing controls; at right an editor window for writing

code. FormWriter also shows the list of user defined procedures as well as the system built-in primitives.

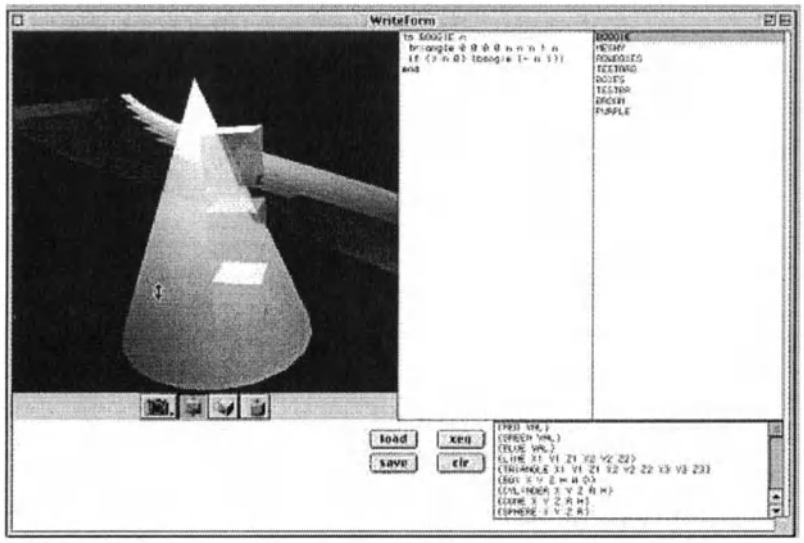


Figure 1. The FormWriter Design Environment

The designer types code in the editor window; to see the result the designer presses the "execute" button. The resulting geometry can be saved as a file and imported into a CAD modeller for further processing. Procedures written in the FormWriter environment can also be saved and reloaded in a future session.

2.1 Simple FormWriter commands

The first thing to do with FormWriter is to generate 3D forms by writing code directly, without defining any procedures. Figure 2 shows the result of a few minutes of play with FormWriter along with the lines of code that generated it. FormWriter's primitive procedures to generate geometry (triangle, cone, box, sphere, cylinder) take dimensions as parameters; the forms are positioned by moving the 3D "flying turtle" forward between the primitive geometry generating procedure calls.

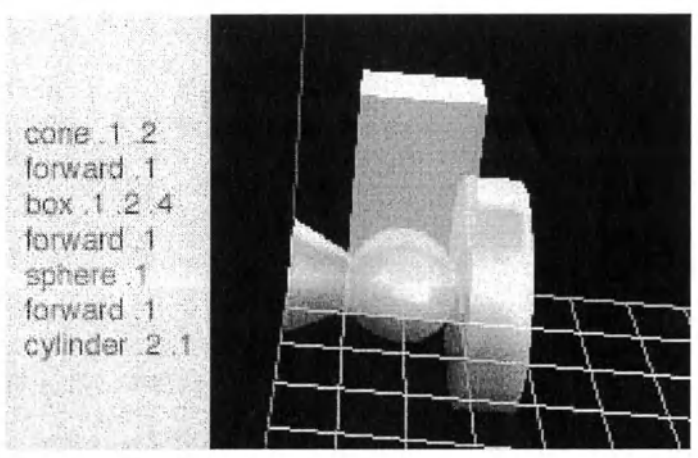


Figure 2. Generating simple forms directly.

Figure 3 shows how the 3D (flying) turtle is used to position and orient five cylinders. The flying turtle can move forward and back, and turn (right and left), pitch (up and down), and roll (side to side). FormWriter inserts each geometric primitive at the current position and orientation of the flying turtle, so as the turtle moves forward (.2 units) and pitches up (30 degrees), between calls to the "cylinder" primitive procedure, each cylinder is translated and rotated in space.

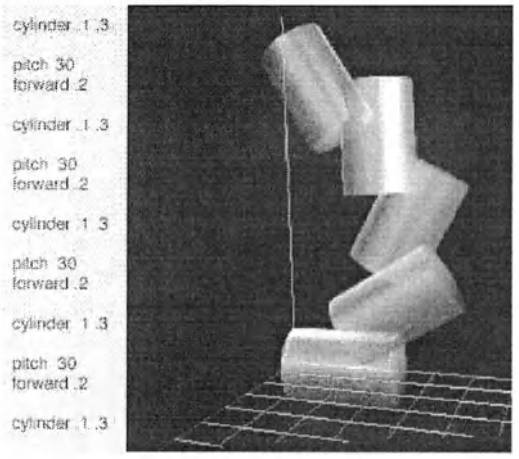


Figure 3. Positioning cylinders in space using the 3D turtle (pitch and forward).

2.2 A first FormWriter procedure

After exploring some simple primitives, and becoming familiar with the flying turtle as a means to position forms in space, the next thing to do is write FormWriter procedures. FormWriter employs the same editor wiindow for defining procedures as for executing commands directly. Each of the forms in Figure 4(a-d) were produced by iterativse calls to the user-defined **one_box** procedure, executing the following line:

repeat (i,10) [one_box()]

Each of the forms used a slightly different definition for the **one_box** procedure as shown in Table 2 below - adding first a turn, then a roll, and finally a pitch instruction to the flying turtle to produce the row of boxes (4a), the turning row(4b), the twisting turning row(4c), and the helix in figure 4(d). The **repeat** statement iterates over a program statement (the call to **one_box**) binding a variable (in this case **i**) to the iteration count.

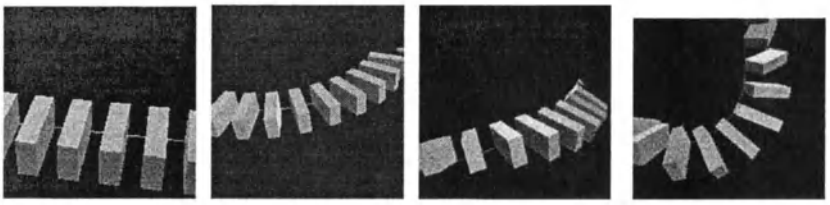


Figure 4. (a) A row of boxes; (b) turning boxes; (c) twisting boxes; (d) boxes helix

Table 2. **one_box** code for the forms in Figure 4

to one_box() box(.4,.1,.2) forward (.2) end	to one_box() box(.4,.1,.2) forward (.2) right(10) end	to one_box() box(.4,.1,.2) forward (.2) right(10) roll(10) end	to one_box() box(.4,.1,.2) forward (.2) right(20) roll(20) pitch(20) end

Figure 5. Triangle variations: Orchid, Spiny Tree, and Cathedral

Figure 5 shows some variations produced by a similar program, this time drawing a triangle and written with an input parameter - the code appears in Table 3. TRI is a simple procedure that sets the display color (red, green, and blue values scaled from 0-1) and traces a triangle polygon from the current position and orientation of the flying turtle. The dimensions of the triangle depend on the parameter N that is passed to the TRI procedure. TRIMANY is a recursive procedures that takes one parameter N, calls TRI with that value (resulting in varying size triangles), and then moves forward a distance depending on N, then rolls, pitches, before calling itself recursively, decrementing N until N goes to zero. Slight variations on this code produced the illustrations in Figure 5.

Table 3. Recursive parametric code for generating triangle forms

```
TO TRIMANY (N)
 TRI(N)
 FORWARD((N/ 20) )
 ROLL(20)
 PITCH(10)
 if (N> 0)  then TRIMANY((N- 1) )
end
```

```
TO TRI (N)
COLOR (.6,.1,.7)
 TRIANGLE(0, 0, N, N, 0, 0)
 COLOR(.1,.7,.3)
 TRIANGLE(0, N, N, N, 0, 0)
end
```

3. LANGUAGE DESIGN

FormWriter avoids the complexities of languages such as C and Java while providing the full computational power of a functional programming language. Three key features of FormWriter make it powerful and easy to use. These are:

1. immediate results, reducing the four-step edit, compile, load, and execute cycle to a 2-step edit and run cycle;
2. variables are not typed; that is, programmers need not declare variables as integer, string, etc.

3. three dimensional space is described differentially, hiding from the programmer the complexities of transformation matrices and coordinate systems.

The language provides a full complement of control structures: iteration (repeat) and conditional execution (if-then-else) as well as recursion; and data types, including numbers, sysmbols, sequences (lists).

3.1 Immediate results

FormWriter provides immediate results: The programmer writes and runs the code in the same environment. Most compiled programming languages such as C and Java require a four step cycle: (1) edit source code; (2) compile source code; (3) load compiled (object) code; (4) execute the program. (Many integrated development environments enable these four steps to take place in a single environment, but these environments are typically quite complex, providing more advanced programming and debugging tools than a novice is prepared to deal with..) One of the earliest novice programming languages, BASIC (Beginner's All-purpose Symbolic Instruction Code), replaced this four step process with a two step integrated edit and execute cycle. BASIC ran as an interpreted language and programmers could write, run, and debug code in a single environment. When the BASIC interpreter detected an error, or when the program did not behave as expected, the programmer could simply retype the offending lines and RUN the program again.

3.2 Untyped variables

The second key feature of FormWriter is that variables are not typed and need not be declared. In C and Java, every variable must be declared as belonging to a specific type (integer, string, array) before it is used in a program. Some other powerful programming languages—Lisp, for example—do not impose type restrictions. Strongly typed languages are generally thought to help programmers write correct code, because this discipline reveals errors at compile time that might otherwise result in obscure run-time behaviour. For a novice programmer, however, the discipline of declaring variable types before using them imposes an additional distance from writing and debugging code.

3.3 Differential (turtle) geometry

The third key feature is that FormWriter programmers use differential geometry—the flying turtle—to perform translations and rotations of scene geometries. This simplifies considerably the deployment of three-

dimensional forms, because the programmer need not keep track of a coordinate system or understand the transformation matrices that most 3D programming libraries offer. Differential geometry for computer graphics was introduced in the novice programming language Logo: It is one of that language's "powerful ideas" and opens the door to interesting experiments in mathematics and graphics (Abelson and diSessa, 1981). The following program to draw a circle reveals the power and simplicity of this approach:

```
to circle
  repeat [forward 1  r ight 1]
end
```

The turtle traces out a circle by repeatedly moving one step forward and turning one degree to the right. By adopting a local frame of reference the programmer needs no recourse to coordinates, trigonometry, or the (cartestian or polar) equation of a circle: only a common-sense experiential knowledge of space.

3.4 Syntax

The careful reader will have noticed some variation in the syntax used in the examples in this paper. We are exploring three alternative syntaxes for FormWriter: a C-like syntax with infix operators, a Lisp syntax, and a Logo-like syntax. Each has its advantages and disadvantages. We currently support all three syntax variations. User programs are first translated into Lisp and then executed by the Macintosh Common Lisp environment.

The C-like syntax will be familiar to anyone who has had contact with a conventional programming language like C or Java. The Lisp syntax has the simplest syntax rules. The Logo syntax is easiest to write for simple programs, but imposes some constraints on functional composition. Table 4 below shows equivalent statements in the three syntax variants.

Table 4. Syntax alternatives for FormWriter

Logo-style	C-style	Lisp-style
red 1	red (1)	(red 1)
to spiral :dist :angle :geom	to spiral(dist, angle, geom)	(to spiral (dist, angle, geom)
pitch :angle	pitch(angle)	(pitch angle)
roll :angle	roll(angle)	(roll angle)
right :angle	right(angle)	(right angle)
forward :dist	forward(dist)	(forward dist)
eval :geometry	eval (geometry)	(eval geometry)
spiral :dist :angle :geom	spiral(dist, angle,geom)	(spiral dist angle geom)
end	end	end)

4. DISCUSSION

A debate has endured for many years as to whether schools teaching computer aided design should teach computer programming. Opponents of teaching programming argue that architects need to learn to use CAD tools effectively in design, and that building software is best left to professional programmers. Proponents of teaching programming argue that it is essential to understand how computers do what they do and to 'open the black box' of computation even for people who will not become professional programmers.

Perhaps the debate is academic. Recently, a surprising number of architecture students—novice programmers *par excellence*— have embraced the coding details of HTML, JavaScript, Lingo, Flash. They want to produce the effects that they can obtain with these languages and to produce these effects they are willing to put up with an amazing amount of complexity.

4.1 Implementation status

The first version of FormWriter is written in Macintosh Common Lisp and uses the QuickDraw3D API to display three-dimensional geometry. The current development version is replacing QuickDraw3D by OpenGL. We are prototyping version of FormWriter written in Java that runs in a browser, producing VRML code that is displayed using a VRML plug-in or Java3D.

5. ACKNOWLEDGEMENTS

The current version FormWriter is written in Macintosh Common Lisp (MCL) and employs "user contributed code" written by Mark Kantrowitz (infix parsing), and John Wiseman (QuickDraw3D). The OpenGL version currently under development employs the OpenGL API written by Alex Repenning of Agentsheets Inc. The Java applet version is being written by Thomas Jung.

6. REFERENCES

Abelson, H. and A. diSessa, 1981, *Turtle Geometry*, MIT Press, Cambridge, MA.

Benjamin, A. and D. Libeskind, 1998, *Reiser + Umemoto : Recent Projects*, John Wiley & Son Ltd, London.

Bentley, J. L., 1986, " Little Languages -- Programming Pearls." *Communications of the ACM.* **29**(August), p. 711-721.

Berzowska, J., 1998, *Algorithmic Expressionism.* Media Laboratory, Cambridge, MA, MIT.

du Boulay, J. B. H., O'Shea, T., and Monk, J., 1981, "The black box inside the glass box. Presenting computing concepts to novices." *International Journal on Man-Machine Studies* **14**(3), p. 237-249.

Kolarevic, B., 2000, "Digital Architectures", In *Proc. ACADIA 2000*, ACADIA, Washington, DC (forthcoming).

Lynn, G., 1998, *Animate Form*, Princeton University Press, Princeton, NJ.

Maeda, J., 1999, *Design By Numbers*, MIT Press, Cambridge, MA.

Papert, S., 1980, *Mindstorms, children, computers and powerful ideas*, Basic Books, New York.

Perrella, S., ed. 1998, *Hypersurface Architecture (Architectural Design Profiles,133)*, John Wiley & Son Ltd;, London.

Terzidis, K., 1999, "Experiments on Morphing Systems", In *III Congreso Iberoamericano de Grafico Digital [SIGRADI Conference Proceedings] Montevideo (Uruguay)*, SIGRADI, p. 149-151.

Toy, M., ed. 1999, *Architects in Cyberspace II (Architectural Design Profile, No 136)*, John Wiley & Son Ltd, London.

Tweed, C., 1986, "A Computing Environment for CAAD Education", In *Teaching and Research Experience with CAAD [4th eCAADe Conference Proceedings]*, eCAADe, p. 136-145.

DesignBUF: Exploring and Extending 2D Boolean Set Operations with Multiple Modes in the Early Design Phase

Jin Won Choi, Do-Young Kwon[1] and Hyun-Soo Lee[2]
Yonsei University, [1]Ajou University,[2]Yonsei University

Key words: Design Buffer, Extended Boolean Set Operations, Structured Floor Plan.

Abstract: Boolean set operations have been a powerful design function set for any CAD systems including 2D and 3D domains. Their capacity to provide even more powerful design tools have not, however, been fully explored in the 2D system. The purpose of this study is to further explore 2D Boolean set operations with multiple modes, which include a pick mode, a wait mode, a drag-and-drop mode, and a draw-and-action mode. We develop a prototype design tool, called DesignBUF. It introduces a new concept of "design object buffer," an intermediate design zone in which a designer freely sketches his/her design with design objects in a brainstorming fashion since valuable design ideas are ephemeral? and the designer needs to generate design schemes rapidly before the ideas disappear or are forgotten. After finishing such fast brainstorming processes, especially in the early design phase, the designer gets a stable and refined form of a floor plan, which in turn becomes a well structured form to maintain building and design information systematically. Therefore, the designer keeps switching back and forth between the "design object buffer" and structured floor plans. We believe that this dual working memory will not only increase system flexibility, but also reduce computation with unnecessarily complex design objects. This study also develops a robust algorithm to transform the intermediate design objects into a well-structured floor plan. In fact, the algorithm is also used for the extended Boolean set operations described above. A structured floor plan can also be transformed into non-structured forms. Research issues for future development are also identified at the end of the paper.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 589–602.
© 2001 *Kluwer Academic Publishers.*

1. THE CREATIVE DESIGN PROCESS

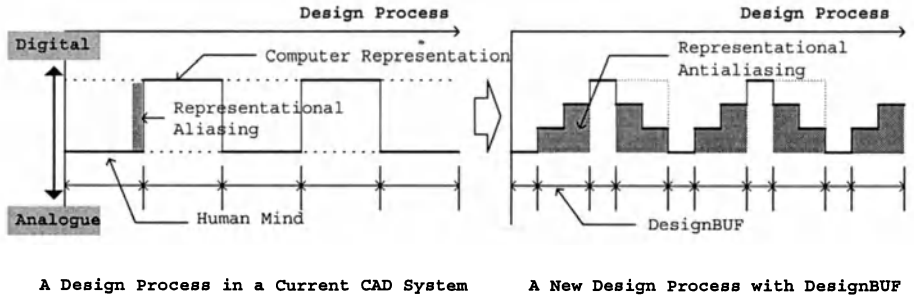


Figure 1. DesignBUF for Representational Antialiasing

A well-known notion is that there are differences between the human mind and computer representations. The differences create a certain gap between how designers think and how computers work. Interfacing these two worlds is critical in developing any design systems. However, most current CAD systems obviously present such aliasing problems (Figure 1). Researchers have attempted various antialiasing strategies, one of which is to allow design flexibility in the early design process. Assume that any architectural design process deals with two different design objects: graphic and architectural. Graphic entities are more appropriate for the early design stages due to their semantically-free and sketch-like attributes, while architectural entities are more adequate for the design development stages due to their semantically-specific and component-based attributes. These two entities often interact and switch back and forth as the design process proceeds. A problem of current CAD systems is the lack of such interactions. It might be important to provide an intermediate design zone wherein two types of entities coexist and can be freely moved back and forth by designers. We call the zone a design buffer or DesignBUF.

Table 1 Graphic Entity versus Architectural Entity

Graphic Entity	Architectural Entity
2D Graphic Systems	Architectural CAD Systems
Semantically-free	Semantically-specific
Not object-oriented	Object-oriented
Suitable for early design stages	Suitable for design development stages
Does not have to be structured	Need to be structured
Can support many CAD theories such as shape grammars, shape emergence, space syntax, etc.	Difficult to support them

DesignBUF also supports a top-down approach, an important design strategy in architectural design. It not only allows designers to develop a design from graphic elements to architectural elements, but also helps designers to develop mass models from? architectural models in a rapid and automated manner.

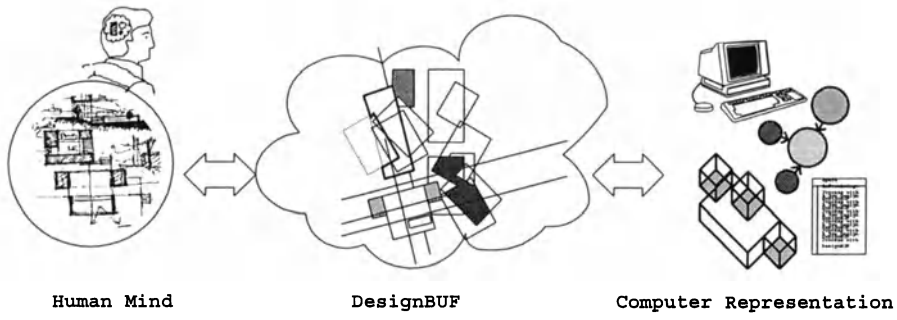


Figure 2. DesignBUF between Human Mind and Computer Representations

2. RELATED WORKS

This study mainly relates to two critical research issues: 1) The form generation process based on mathematical rules and enhanced user interactions, and 2) Space-based representations.

2.1 The Form Generation Process

Shape emergence is a well-known research issue in the form generation process. Tan (1990) discusses the centrality of emergent forms in the design process and proposes a data structure based on construction lines and ordered lists which enables shapes as a collection of lines and arcs to be efficiently encoded. Shape grammars obviously relate to emergence issues in terms of the continuity of computations (Stiny, 1993).

ReDraw (Kolarevic, 1994) is a computer-based graphic environment for design conceptualisation, especially for shape delineation and dynamic drawing manipulation, and is based on construction lines and their geometric relations.

Chase (1996) describes a way of using shape algebras and emergence instead of more traditional representations of geometric objects. His method attempts to describe designs combining the paradigms of shape algebras and predicate logic representations.

He (1999) also investigates better user interaction modes in the grammar-based design process. He proposes possible modes of user interaction and control within grammar-based design systems.

2.2 Space-based Representations

There has been some research on developing CAD systems that can build structured floor plans (Yessios, 1986a & 1986b; Choi, 1997; Kalay et al, 1998). In such systems, a floor plan is structurally well defined by having some hierarchical components. Since space and form are the two main aspects to define a building, they should be represented at the same time within a system.

A space-based representation is also examined in Mahdavi's recent research (1999). Trying to integrate detailed simulation methods and CAD systems, he recognized the importance of spatial information. He observed that detailed thermal simulation methods require the definition of spaces and zones, and not just bounding surfaces.

Carrara's research (1994) focuses on using spatial information in the very first stage of the design process. The concepts of Space Units (SU) and Building Units (BU) that the researchers employed allowed the system to represent the defined SUs as bubbles, highlighting the adjacencies and the defined paths.

An interactive graphic approach (Liggett, 1992) has been attempted to solve spatial allocation problems in facility layout. The study introduced the concepts of 'stack plan', 'zone plan', 'block plan' and '1to1 plan'. To generate an ideal block plan, a designer employed a process of generation, adding/modifying criteria, and appropriate trade-offs in an iterative, interactive fashion.

3. EXTENDING BOOLEAN SET OPERATIONS

3.1 Boolean Set Operations

Boolean set operations have been a powerful design function set for any CAD systems including 2D and 3D domains. They are particularly useful operations in any solid modelling system. Some general types of Boolean set operations include union, intersection, difference, and split. The general algorithm of each operation defines where to start tracing and how to trace line edges. For example, the algorithm of the union operation can be described as follows:

1. *Start from a vertex on the first object where the starting vertex is located at the outside of the intersected area.*
2. *Trace line edges in the positive direction until an intersection vertex encounter.*
3. *Jump to the other object and trace line edges in the positive direction until an intersection vertex encounters.*
4. *Repeat 2 and 3, and finish the process if it reaches the starting vertex again.*

While the operations are simple and clear to use in any form generation stages, they have not yet been fully explored to provide even more powerful design tools, especially in 2D architectural CAD systems. One of the most critical limits is that the operations deal with only two objects at a time. This could be a problem when a designer deals with many design objects, wants to figure out the relationships among the objects, and tries to find derivative forms based on the objects. The interaction mode is also limited even though some different mode types can be considered.

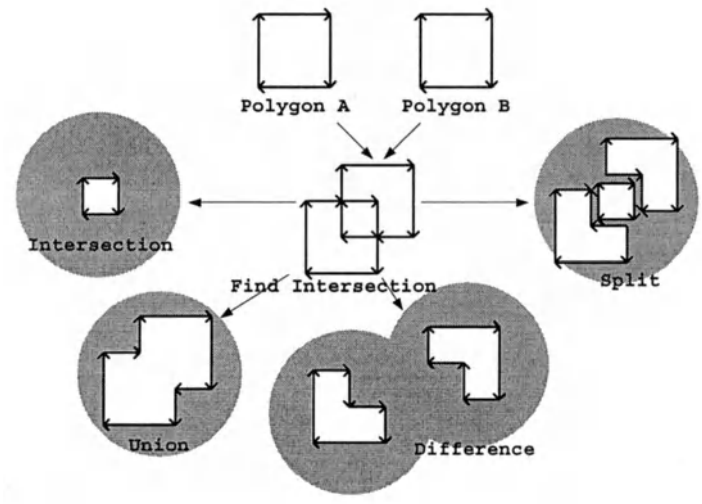


Figure 3. Boolean Set Operations

3.2 An Extension of Boolean Set Operations

As the term "Boolean" denotes, the Boolean set operations deal with only two objects. While the concept is simple and clear to understand, it often limits designers to develop various forms in a creative fashion.

The extended version of the Boolean set operations in this study can thus deal with multiple objects simultaneously. Along with the original function

set including union, intersection, difference, and split, a new function type, called 'offset, ' adds its power. The offset function converts? a line or a polygon to a polygon object with a width. The function sets we developed also work with two types of objects, void and solid. If we consider the solid type as a normal object, the void object is a new type where the line edge information is important and there is no spatial information related to the object. The sets also work with multiple modes in terms of user interactions.

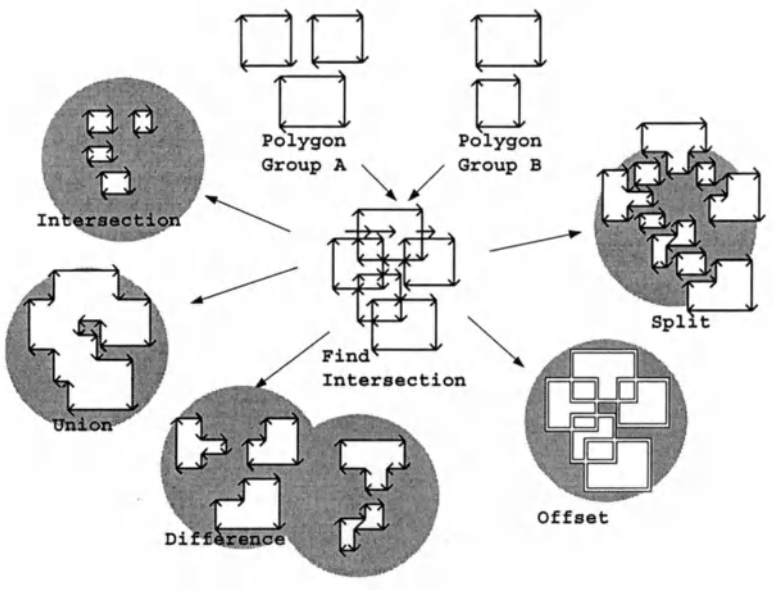


Figure 4. Extended Boolean Set Operations

3.3 Void and Solid Boolean Set Operations

If we assume that current 2D Boolean set operations only deal with solid objects filled up with something, we can consider void objects where nothing is inside. A solid object has a selectable area, filled with a user-defined colour. On the contrary, a void object has a transparent area. Operations work with objects of the same type. That means solid objects only work with solids. The following figure shows how two types of operations result in different outputs.

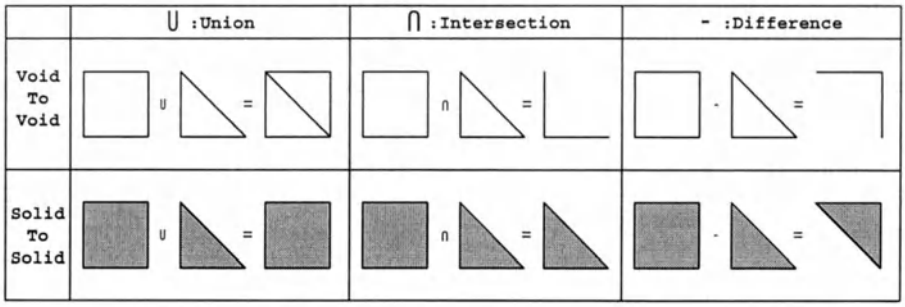


Figure 5. Types of Solid and Void Boolean Set Operations

The following figure presents a process using extended Boolean set operations. The process starts with creating polygons, followed by structuring polygons. The next step is to specify spatial information. The outputs can be represented in different ways: void/solid objects, a space diagram, 3D models, a floor plan, etc.

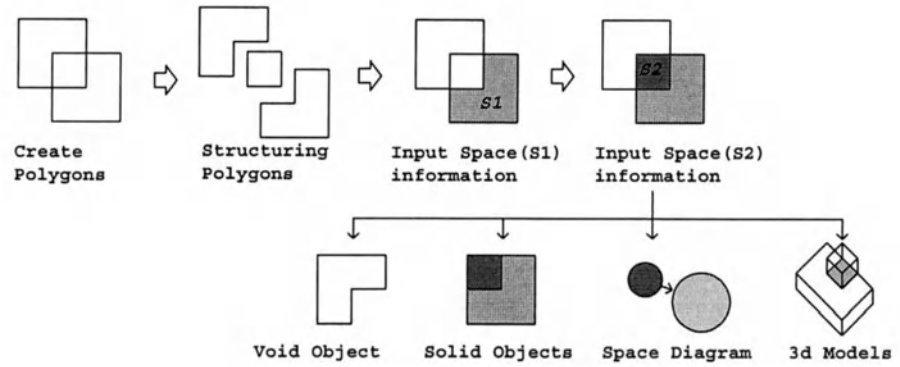


Figure 6. A Process Using Solid and Void Boolean Set Operations

3.4 Multiple Modes

Four types of multiple modes are based on the action point of the time line in the early design phase. A pick mode, a first among four modes of the extended Boolean set operations, is a normal mode shown in any other CAD system. The designer needs to pick two design objects to conduct a Boolean operation. In a wait mode, the system waits until the designer asks to conduct a Boolean operation after finishing a set of editing operations, such as transformation and rotation. Further, a drag-and-drop mode allows the designer to simply drag and drop design objects onto other objects in order to activate a given Boolean operation. This mode works well with the

symbol and block concepts. Using symbol/block databases, one can design many alternate design solutions. And finally, in a draw-and-action mode the designer draws an object on existing objects and the system automatically processes a Boolean operation. The designer can work in any mode among those four during any moment of the design process.

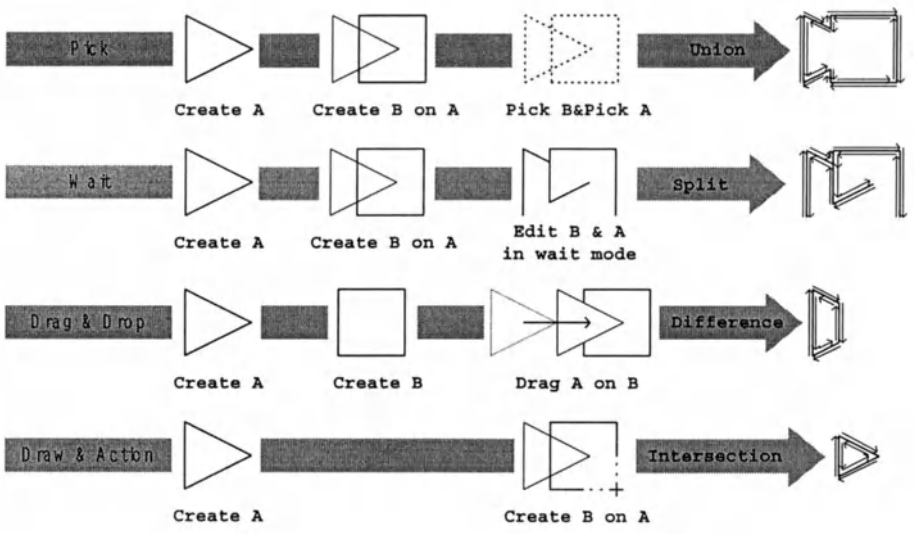


Figure 7. Extended Boolean Set Operations with Multiple Modes

4. THE BASIC ALGORITHM FOR EXTENDED BOOLEAN SET OPERATIONS

4.1 Basic Strategies in The Process

The following figure presents how original objects are divided into multiple spatial polygons that have relationships with each other. Overlapped objects (A and B) generate intersected areas(C). They also construct a set of spaces including an outside space representing a boundary of the total objects. We have developed? an algorithm that automatically structures and represents the spaces included.

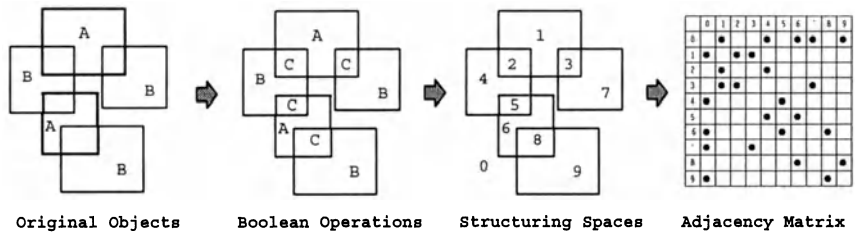


Figure 8. The Process of Extended Boolean Set Operations

The algorithm for extended Boolean set operations focuses on dealing with multiple objects at the same time and structuring spatial information automatically. The basic strategies are, therefore, as follows:

1. The input geometry is composed of a set of several input lines that may or may not have intersections with the existing geometries and even self-intersections with the input lines themselves. In Figure 10, input points 1 to 7 are a set of input geometry.
2. A set of input geometry is instantly processed to maintain a structured form.
3. A set of input geometry is divided into several units, taking into account intersections with existing geometries and self-intersections. A geometrical unit, therefore, does not have any intersections. Figure 10 shows five geometrical units divided.
4. There are four types of intersections with existing geometries: T, Y, I and N.
5. A special consideration is required in case both ending points (a starting point and a target point) of a geometrical unit are located on an identical existing geometry.
6. A special consideration is also needed with inputs paralleled and attached to existing geometries.
7. Existing rings through which ending points pass should be recognized. A special consideration is required when the existing rings differ from each other.
8. Vectors of geometrical surfaces should be rerouted in a defined fashion based on the intersection type.
   Following is a description of the process in detail.

4.2 The Break-Down of The Input Geometry

Input geometries are a sequence of input lines given by the designer and allows self-intersections with the input lines themselves. For the simplicity

of the creation algorithm, these input lines need to be broken down into a set of geometrical units with no intersections at all.

The example in Figure 10 presents complex input geometry on existing geometries. The breakdown process divides the input geometry into five units. The following creation process is repeated five times to finish the whole process.

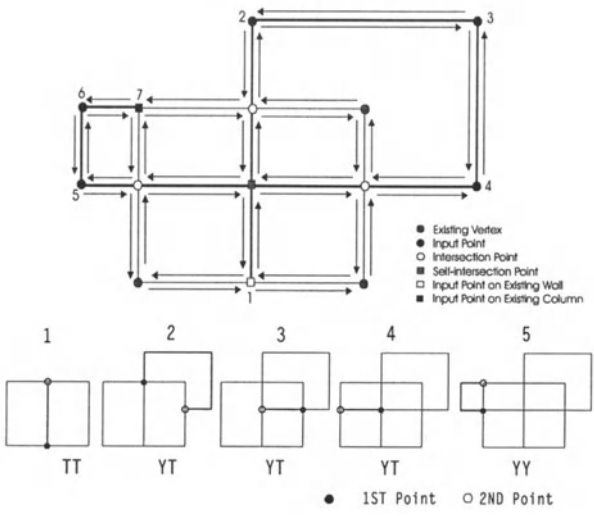
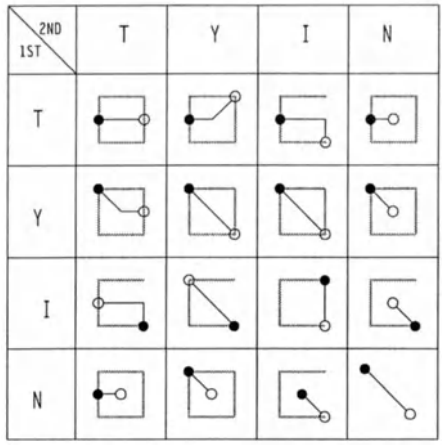


Figure 9. The breakdown process of the input geometry

4.3 Intersection Types in a Geometry

Each point in a geometrical unit, having two end points, a starting point and a target point, has one of four intersection types such as T, Y, I or N. Type T occurs when the input point is located on an existing geometry, while the input point of Type Y is on an existing column where two existing geometries are crossing one another. In Type I, the input point is placed on the end of an existing geometry where two geometries are not crossing. The input point on Type N is freely located on a ring detached from any geometry.

Table 2 Intersection Types in a geometry unit



4.4 Rerouting Geometry Surface Vectors

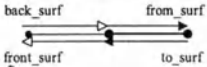
Once intersection types are recognized, vectors of geometrical surfaces connected to the input points are rerouted to maintain a correct structure. The rerouting process occurs in three intersection types excluding Type N.

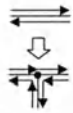
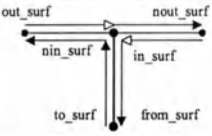
First, a relatively simple rerouting process occurs for Type I. When from_surf and to_surf are geometry surfaces belonging to the input line, back_surf is connected to from_surf and to_surf to from_surf.

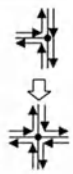
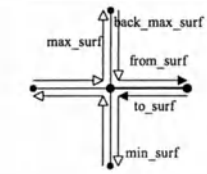
Next, the process for Type T makes one wall and two geometry surfaces, nout_surf and nin_surf.

Lastly, for Type Y it is important to find two vectors, one closest to the input line in a clockwise direction and the other in a counter-clockwise direction. To do this, it is necessary to calculate angles between each geometry and the input line after collecting all surface vectors connected to the vertex.

Table 3 Rerouting wall surface vectors for intersection types

Type	Scheme	Rerouting Process	Pseudo-code
Type I			back_surf->next = from_surf; to_surf->next = front_surf;

Type T			out_surf->next = nout_surf;
			in_surf->next = from_surf;
			to_surf->next = nin_surf;

Type Y			to_surf->next = min_surf; back_max_surf->next = from_surf;

5. CONCLUSIONS AND FUTURE RESEARCH

5.1 Conclusions

Based on the notion of the gap between the way designers work and the way computers represent, this study explores a way of increasing design flexibility. The method we have developed? is based on an extended set of 2D Boolean set operations which, we believe, allow designers to start with some graphical design elements, develop emergent forms using such operations, and end with a well developed model with architectural elements generated from the semantically-free forms previously generated?. The method introduces void objects opposite to existing solid objects. Considering these two types of objects instead of only one would give more chances for designers to think and develop forms in different ways. Multiple user interaction modes also add flexibility since designers can delay or process their operational decisions at any time in the design process. As an end result, this set of operations would add diversity in terms of functionality if applied in any architectural CAD system.

In fact, this not only increases design flexibility, but also improves design productivity since the method integrates 2D and 3D design environments. That is, it automatically generates 3D architectural models based on 2D floor plan data.

This method would allow designers to evaluate building performance from various perspectives. It builds a structurally well-defined floor plan that has information about spaces within the floor plan and their relational

information. Based on such information, various building performance evaluation is possible when individual evaluation modules are developed.

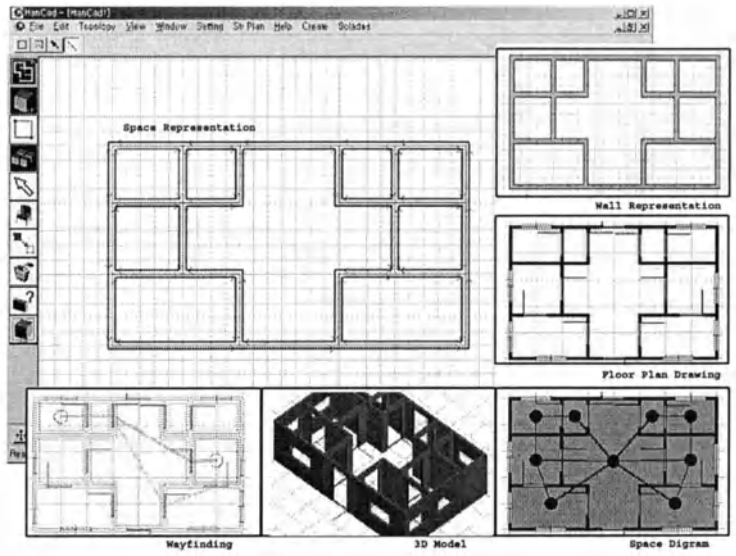


Figure 10. The Use of DesignBUF

5.2 Future Research

The system, currently under development, has two different versions: a standalone version written by C++ and a Java applet version. The Java applet also includes other research issues as a tool for a collaborative design.

Further research issues include developing a mechanism to efficiently record design changes as a design evolves and move from one design to the other with a set of recorded designs.

The approach applied in this study, mainly based on a space-based floor plan representation of buildings, has extensive potential application areas. Although some of them are partly implemented and tested in the DesignBUF environment, many others still need to be explored. For example, based on spatial information, DesignBUF can automatically construct floor plans with 3D building information that in turn generate a 3D solid model. Beyond that, we are exploring a way of managing a floor plan and a 3D model simultaneously. When a change is caused in a floor plan, another corresponding operation is transparently activated to adjust the 3D model.

6. REFERENCES

Branko Kolarevic: 1994, Lines, Relations, Drawing and Design, *Reconnecting,1994 ACADIA Proceedings*, p.51-61

Carrara, G. & Kalay Y.: 1994, Knowledge-based computational support for architectural design, *Automation in Construction*, 3, p.157-175

Choi, J.W.: 1997, A development of an intelligent CAD engine to support architectural design collaboration, *Korea CAD/CAM Journal*, p.53-59.

Eastman, C.M. et al.: 1991, *A Data Model for Design Databases, in* J.S. Gero (ed), Artificial Intelligence in Design '91, Butterworth- Heinemann, Oxford.

Eastman, C.M.: 1991, A Data Model Analysis of Modularity and Extensibility in Building Database, *Environment and Building*, 27(2), pp.135-148.

Eastman, C.M.: 1994, A Data Model for Design Knowledge, *Automation in Construction*, 3, pp.135-147.

George Stiny: 1993, Emergence and Continuity in Shape Grammars, *CAAD Futures ' 93*, p.37-54

Kalay, Y, Khemlani, L, & Choi, J.W.: 1998, An Integrated Model to Support Distributed Collaborative Design of Buildings, *Automation in Construction, 7*, pp.177-188.

Kim, U.: 1992, An Object-based Modeling System Coupled with Design Information System, *6th International Conference on Systems Research*, Baden Baden, Germany.

Kim, U.: 1995, *An Object-based Building Product Model : From Modeling to Documentation*, International Council for Building Research Studies and Documentation, W78 TG10 Workshop, Stanford, U.S.A.

Liggett, R.S.: 1992, *Designer-automated algorithm partnership: an interactive graphic approach to facility layout, in* Kalay, Y. (ed), Principles of Computer-Aided Design: Evaluating and Predicting Design Performance, John Wiley & Sons, Inc.

Mahdavi, A.: 1999, A comprehensive and computational environment for performance based reasoning in building design and evaluation, *Automation in Construction*, 8, pp.427-435

Maher, M.L. & Simoff, S.J.: 1997, Formalising building requirements using Activity/ Space, *Automation in Construction*, 6, pp.77-95

Milton Tan: 1990, Closing in an Open Problem – reasons nad a strategy to encode emergent subshapes, *1990 ACADIA Proceedings*, p.5-19

O'Neill M.J.: 1992, *Neural network simulation as a computer-aided design tool for predicting wayfinding performance, in* Kalay, Y. (ed), Principles of Computer-Aided Design: Evaluating and Predicting Design Performance, John Wiley & Sons, Inc.

Scott C. Chase: 1996, Design Modeling With Shape Algebras and Formal Logic, *Design Computation, 1996 ACADIA Proceedings*, p.99-113

Scott C. Chase: 1999, Grammar Based Design: Issues for User Interaction Models, *Media and Design Process, 1999 ACADIA Proceedings,* p.198-210

Steadman, P.: 1993, A database of the non -domestic building stock of Britain, *Automation in Construction*, 2 (1) pp.31-45.

Yessios, C.I.: 1986a, *The Computability of Void Architectural Modeling*, The Computability of Design, Proceedings of Symposium on Computer-Aided Design at Suny Buffalo, New York.

Yessios, C.I.: 1986b, *Architectural Modeling, Architecture 844 Lecture Notes I*, Graduate Program in Computer-Aided Architectural Design, Department of Architecture, The Ohio State University.

POCHE'

Polyhedral Objects Controlled by Heteromorphic Effectors

Ganapathy Mahalingam
North Dakota State University

Key words: Effectors, **abstract** machines, design as interface

Abstract: This paper takes the architectural concept of poche' and uses it to explore new possibilities in transforming polyhedra with effectors. In many computer-aided design systems, architectural entities are represented as well-formed polyhedra. Parameters and functions can be used to modify the forms of these polyhedra. For example, a cuboid can be transformed by changing its length, breadth and height, which are its parameters. In a more complex example, a polyhedron can be transformed by a set of user-defined functions, which control its vertices, edges and faces. These parameters and functions can further be embodied as effectors that control and transform the polyhedra in extremely complex ways. An effector is an entity, which has a transforming effect on another entity or system. An effector is more complex than a parameter or function. An effector can be a modelled as a virtual computer. Effectors can take on many roles that range from geometric transformation agents and constraints to performance criteria. The concept of the poche', made famous by Venturi is familiar to architects. The poche' is a device to mediate the differences between an interior and an exterior condition or between two interior conditions. In a poche', the role of the effector changes from being an agent that acts on a polyhedron from the outside, to an agent that acts as a mediator between an interior polyhedron and an exterior polyhedron, which represent interior and exterior environments respectively. This bi-directionality in the role of the effector allows a wide range of architectural responses to be modelled. The effector then becomes an interface in the true sense of the word. This concept will work best in a three-dimensional or four-dimensional representational world but can be used effectively in a two-dimensional representational world as well. The application of this concept in design systems is explored with examples drawn from the work of the author, and practitioners who are using the concept of effectors in their work. A brief discussion of how this technique can evolve in the future is presented.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 603–614.
© 2001 *Kluwer Academic Publishers.*

1. ARCHITECTURAL ENTITIES AS POLYHEDRAL OBJECTS

Architectural entities can be classified as physical or conceptual. Physical architectural entities are made of building materials and have geometric form. Physical architectural entities comprise building materials, components and assemblies. Conceptual architectural entities may have geometric form but are not made up of any material. Conceptual architectural entities comprise entities such as ordering systems and circulation systems. Both physical and conceptual architectural entities have spatial location. Conceptual architectural entities can influence the geometric form and spatial location of physical architectural entities. Conceptual architectural entities, in turn, can be defined by physical architectural entities. Architectural design can be considered as the definition and integration of physical and conceptual entities and fixing their location in space.

In computer-aided design systems, physical architectural entities are represented as well-formed polyhedra. Polyhedra, as their name implies, are volumes defined by a closed boundary of faces. The representation of architectural entities as well-formed polyhedra is called boundary representation. By their definition polyhedra are finite and can be fabricated with material. In their computer-based representation, polyhedra are defined as hierarchical collections of vertices, edges and faces. Of these, the actual variables are the values for the tuples that define each of the vertices of a polyhedron. These variables, in turn, determine the variations in the dimensions and spatial locations of the edges and faces of the polyhedron. Controlling the variables allows a designer to control the various forms that the polyhedra can take. The manipulation of form, which is one of the principal activities of the designer, can be enhanced by creating armatures for the manipulation and transformation of architectural entities represented as polyhedra. The concept of effectors provides one such armature.

2. EFFECTORS

An effector is an entity, which has a transforming effect on another entity or system. An effector is more complex than a constraint, parameter or function. An effector can be a computational entity in its own right. It can accept different kinds of input, perform computations and cause an effect in another entity as part of its output. An effector can be modelled as a virtual computer that can embody both state and behavior (Mahalingam, 1998). Effectors can take on many roles that range from geometric transformation

agents and constraints to performance criteria. Multiple effectors can be integrated as a network of effectors that act in a concerted manner to function as a meta-effector. A hierarchical system of effectors and meta-effectors can also be developed that may or may not be concerted in their transforming effects. The overall effect of a hierarchy of effectors, that do not act in a concerted manner, needs to be controlled by a meta-effector.

Effectors change other entities based on computations. Effectors are different from constraints or parameters. Constraints specify an acceptable range of values for a variable, or relationships between variables. When a constraint specifies a maximal or minimal value for a variable, then it is considered a "limit." If the constraint specifies a range of values for a variable that straddles a certain value, then it is considered a "tolerance." The acceptable values for a variable specified by a constraint can be simply declared, or computed based on a function of that variable or other associated variables.

A parameter specifies values for a variable indirectly. The actual values of the variables are computed using the parameter. For example, the parameters for a cuboid are length, breadth and height. Changing these parameters changes the values for the vertices of the cuboid. The values of the vertices are computed using the parameters. A parameter needs an origin in order to compute its effect unambiguously. For example, changing the length of a cuboid can change only four or all eight of its vertices, depending on whether a vertex or the centroid of the cuboid is used as the origin.

A function can also be used to determine the value of a variable, which is usually called the dependent variable. A function can be any mathematical relation and may not require an origin to compute its effect. A function is defined in terms of one or more independent variables. The function may sometimes be recursive and use the variable it is determining in its computation. Effectors subsume constraints, parameters and functions when they are modelled as virtual computers.

When an effector is modeled as a virtual computer, a network of effectors becomes a network of computers. Polyhedra controlled by bi-directional effectors become a multi-layered network. Mathematical models used to model neural networks, parallel systems and 3D graphs are all viable tools to model networks of effectors. In its most abstract and general form, an effector is a relation between two virtual computers. Each virtual computer can change the state and behaviour of the other virtual computer. This relationship can be realised through one or several computational processes.

2.1 Homomorphic and heteromorphic effectors

If the component of a polyhedron being controlled by an effector is the same kind as the effector, then the effector is homomorphic. If the effector is not the same kind as the component it is transforming, then the effector is heteromorphic. To determine if an effector is the same kind as the component it is transforming, it should correspond in form or type. Geometric transformation effectors transforming the geometry of a polyhedron are homomorphic effectors. This is type correspondence. Since effectors are not geometric entities, the issue of form correspondence arises only if there are connected effectors. For example, two effectors joined to form a straight line that transform the edges of a polyhedron are considered homomorphic effectors. Also, a polyhedral network of effectors transforming a polyhedron is considered a homomorphic effector.

3. POCHE'

The concept of the poche', made famous by the architect Robert Venturi, is familiar to architects. The poche' is a device to mediate the differences between an interior and an exterior condition or between two interior conditions. It is usually used to resolve two conflicting requirements or conditions.

In his influential book, *Complexity and Contradiction in Architecture*, Venturi explains the wide ranging implication of the concept of poche' by quoting Gyorgy Kepes, "Every phenomenon - a physical object, an organic form, a feeling, a thought, our group life - owes its shape and character to the duel between opposing tendencies; a physical configuration is a product of the duel between native constitution and outside environment." (Venturi, 1966). Venturi also states that designing from the outside in, as well as the inside out, creates necessary tensions, which help make architecture. He goes on to make a significant and influential statement, "Since the inside is different from the outside, the wall - the point of change - becomes an architectural event." The concept of bi-directional effectors takes the condition of poche' described by Venturi and provides a computational framework to implement the design processes that he describes. For example, the "native constitution" of an entity can be governed by uni-directional effectors, and "duel" with the "outside environment" can be mediated by bi-directional effectors.

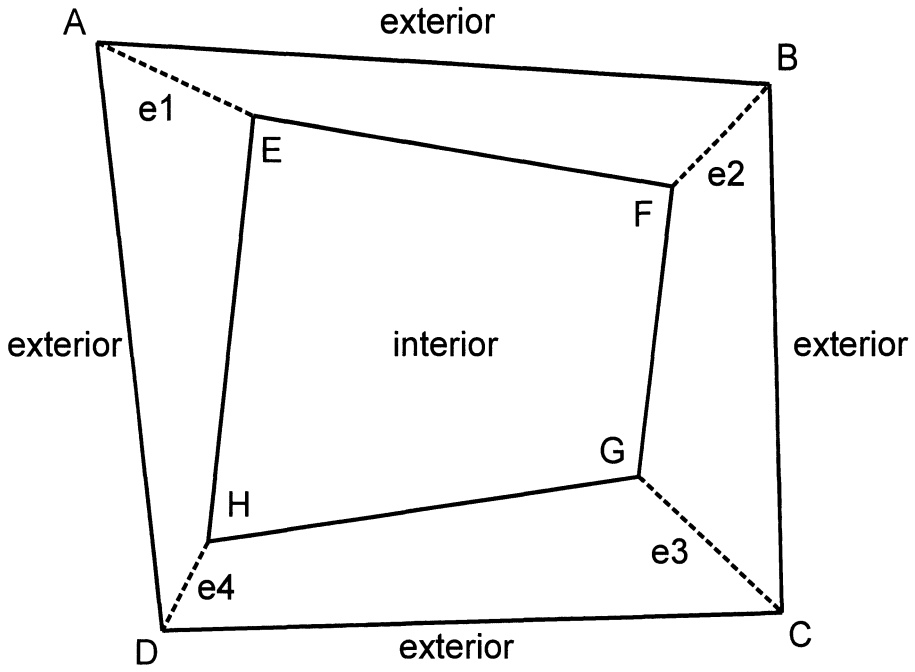


Figure 1. A simple poche' condition with four bi-directional effectors in a two-dimensional representational world

In Figure 1, an example of a poche' condition in a two-dimensional representational world is shown. This figure is not a literal diagram but represents an abstract machine in the Deleuzian sense (Deleuze and Guattari, 1987). ABCD is an exterior polygon and EFGH is an interior polygon. The states of A, B, C and D are determined by exterior contextual criteria. The states of E, F, G and H are determined by interior performance criteria. The four bi-directional effectors are e1 (AE), e2 (BF), e3 (CG) and e4 (DH). The bi-directional effectors e1, e2, e3 and e4 can be parameters, constraints, functions or virtual computers (Mahalingam, 1998). The role of the bi-directional effectors is to mediate and determine the states of the interior/exterior variable pairs that they link. The four bi-directional effectors can be linked to form a polygon of effectors thereby defining a meta-effector that is homomorphic. A meta-effector is a network of effectors. The role of the meta-effector in this case is to co-ordinate the effects of the four bi-directional effectors. Meta-effectors can be constraining agents. If, for example, the above condition represents a single-room structure, then the meta-effector can ensure that a minimum wall thickness is maintained.

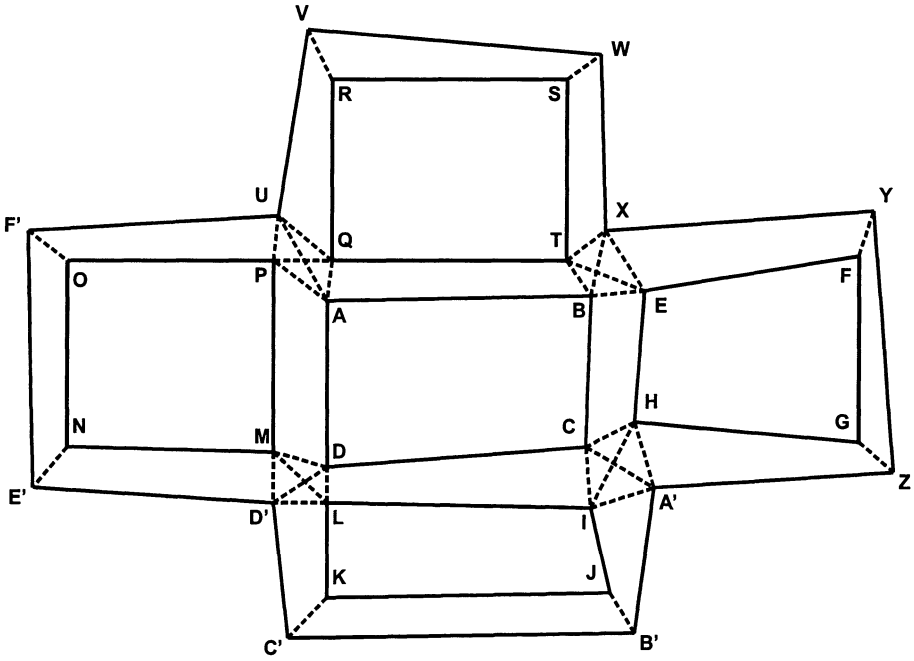


Figure 2. A more complex poche' condition with bi-directional effectors in a two-dimensional representational world

In Figure 2, a more complex, multicellular poche' condition is shown in a two-dimensional representational world. This figure is also not a literal diagram but represents an abstract machine in the Deleuzian sense (Deleuze and Guattari, 1987). There are five interior polygons and an exterior polygon. There are eight peripheral bi-directional effectors and there are four clusters of bi-directional effectors in the interior, each cluster forming a nexus of effectors affecting four variables. In each nexus the state of each one of the four variables needs to be resolved based on the simultaneous effects of the individual effectors. Each nexus can be modeled as a meta-effector. The four clusters that each forms a nexus can be networked as a polygon to create another meta-effector that is one step higher in a hierarchy of effectors. One role of the "nexus" meta-efffectors is conflict resolution at each nexus, to resolve conditions such as spatial overlap. A "nexus" meta-effector can also be used to maintain a minimum separation distance between the variables. Alternatively, a "nexus" meta-effector can collapse into a simple bi-directional effector, suggesting a merging of some of the cells in the multicellular configuration.

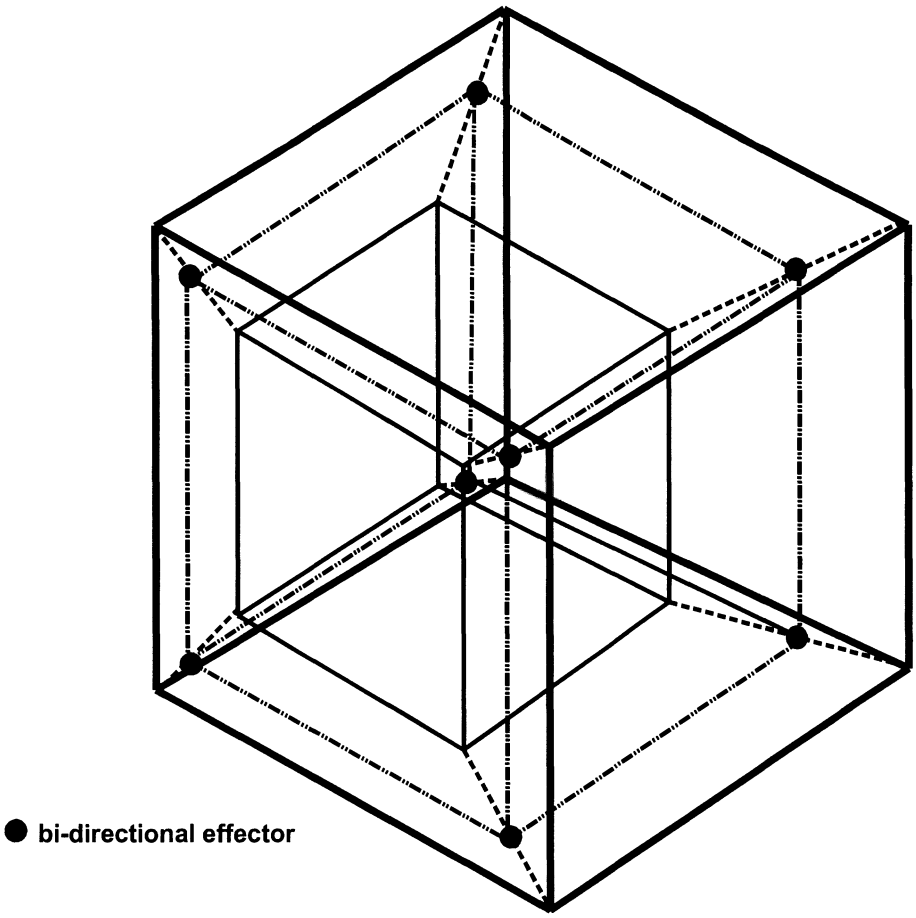


Figure 3. A simple poche' condition with bi-directional effectors in a three-dimensional representational world

In Figure 3, a model consisting of three nested polyhedra is shown. If the polyhedron in the middle (shown in the dash-dot line) is a network of effectors, and the polyhedra on the inside and outside represent interior and exterior environments, then the whole system represents the condition of poche' in a three-dimensional representational world. The role of the effector changes from being an agent that acts on a polyhedron from the outside, to an agent that acts as a mediator between an interior polyhedron and an exterior polyhedron, which represent interior and exterior conditions respectively. This bi-directionality in the role of the effector allows a wide range of architectural responses to be modelled, especially simultaneous responses to interior performance criteria and external contextual conditions. The bi-directional effector can be a mediating channel through

which conflicting conditions between interior and exterior environments are resolved (mediated?). The bi-directional effector then becomes an interface in the true sense of the word.

3.1 Application of effectors in architectural design

The application of effectors in digital design processes for architecture holds a lot of potential. Since architectural entities are usually represented as polyhedra, the transformation of the polyhedra by effectors becomes a central part of design processes that shape forms and space.

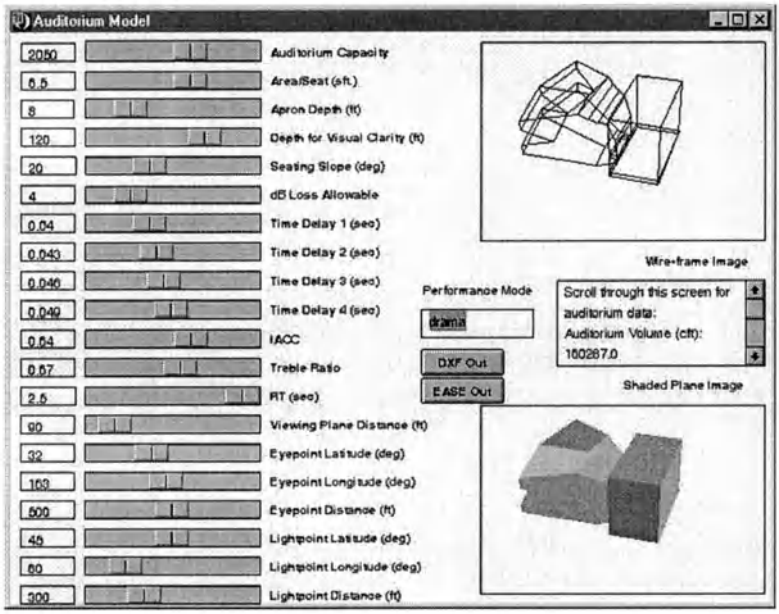


Figure 4. View of auditorium design system developed by the author

The author has developed an auditorium design system (see Fig. 4) where the polyhedral form of a proscenium-type auditorium is generated based on multiple functions of acoustical, programmatic and functional parameters (Mahalingam, 1996). The functions that locate the vertices of the polyhedra that make up the auditorium can be likened to uni-directional effectors. These effectors take the role of functions. The model in the auditorium design system that was developed was inwardly oriented. The design system can now be extended with bi-directional effectors to control an exterior polyhedral form that responds to external site conditions.

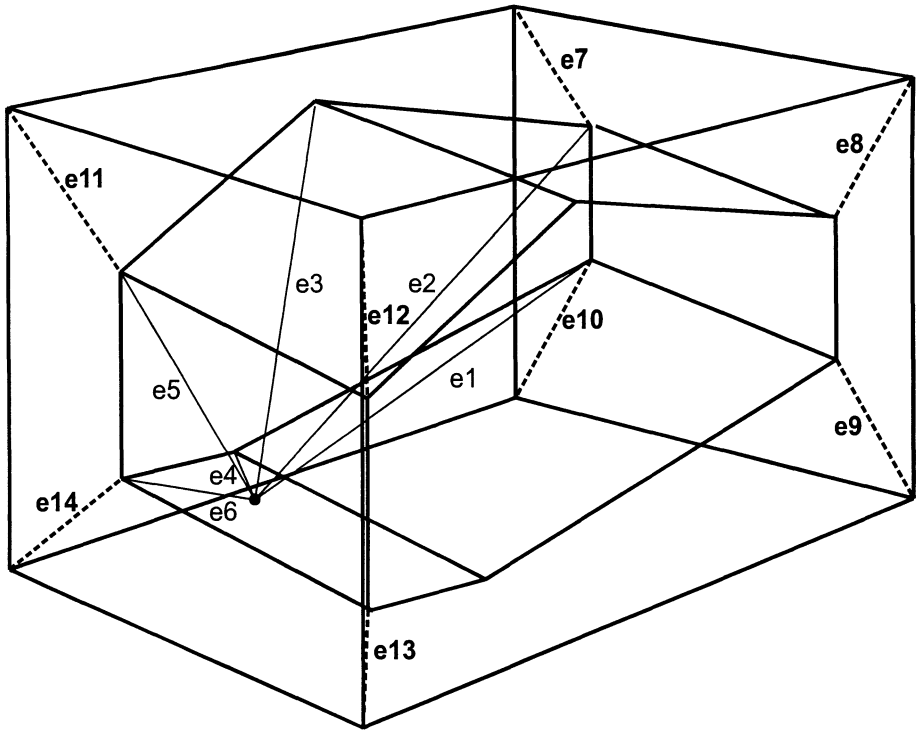


Figure 5. A conceptual diagram of the integration of the concept of effectors in the auditorium design system

In Figure 5, the spatial form of an auditorium seating enclosure is shown. The vertices of the polyhedron that make up the seating enclosure is determined by a set of twelve uni-directional effectors (e1...e6 represent one half of a symmetrical set of twelve effectors). The exterior polyhedron of the auditorium is governed by eight bi-directional effectors (e7...e14). Based on the mediating action of the bi-directional effectors, the exterior polyhedron can respond to site and other contextual conditions, while at the same time responding to the interior polyhedron that is generated by the interior performance criteria. This is just a conceptual example. The complexity of the exterior polyhedron can be increased to address the complexity that a particular contextual condition demands.

3.2 Application of effectors in architectural practice

Recently, a number of projects have been published that have used dynamic and non-linear computational processes to generate architectural designs. A recent issue of *Architectural Design* focusing on contemporary processes in architecture features two projects that can be examined for their relationship to the concept of poche'.

An outstanding project featured in the issue (Architectural Design, 2000), Embryologic Houses by Greg Lynn, uses the concept of effectors, but in a limited way. Lynn describes the underlying concept of his Embryologic Houses thus, "the variations in specific house designs are sponsored by the subsistence of a generic envelope of potential shape, alignment, adjacency and size between a fixed collection of elements." This generic envelope that is subject to mutation is composed of 2048 panels, 9 steel frames and 72 aluminium struts defining a shell. The form and space of the houses are modified within the predefined limits of the components. This is analogous to a polyhedron with a fixed set of vertices, edge, faces and constraints. All the effectors (transforming agents or control points) in Lynn's project act on the generic envelope from the outside and do not mediate between an "interior" and an "exterior" requirement. In fact, the variations in the houses are described as an adaptation to "contingencies" of lifestyle, site, climate, construction methods, materials, spatial effects, functional needs and special aesthetic effects. In the prototyping stage, six houses were developed exhibiting a unique range of domestic, spatial, functional, aesthetic and lifestyle "constraints." How these "contingencies" and "constraints" affect the generic envelope is not clearly articulated, so their role in generating the design cannot be determined. The transforming agents that mutate the generic envelope are causal agents and not mediating agents.

In another project featured in the same issue, Ali Rahim describes the operational principles in the generation of his competition winning entry for a Steel Museum in South Asia thus, "This (abstract machine) was comprised of vectors, fields, pressures and constraints in combination with inverse kinematics, particles and surfaces, embedded within the confluence of virtual matrices." Rahim's abstract machine, or machinic phylum as he prefers to call it, involves the causal transforming effect of particles and vectors (the flock of contaminants) on virtual matrices (the unactivated field) and vice versa. The interaction between the two enables the design. Though there is an exchange between the particles and the virtual matrices, again there is no mediation between an "interior" an "exterior" requirement. The actual spaces for the accommodation of program elements emerge from fluctuating intensities that indicate spatial potential. These intensities result from the indeterminate interaction between the particles and the matrices. The process is akin to crystallisation and annealing, or an act of congealing

or solidification. Duration and temporal evolution determine space. There are no desiring, inhabiting forces that create spatial potential, like the need for a pleasant acoustical environment, thermal delight, or a view to the outside. There is a significant absence of interiority in the design generation processes that both these projects use, even though there is a creation of interiors in both of them. The forms and spaces created provide "opportunities" for occupancy, but there are no active occupying or dwelling forces that generate the forms and spaces. The concept of poche' forces attention on this "interiority."

3.3 Future directions

With growing attention being focused on digital processes for architectural design, a well-defined mechanism, or an abstract machine that triumphs over mechanisms (Deleuze, 1988), needs to be developed to generate the process space for these design processes. The concept of effectors provides one such mechanism/machine. Effectors can be configured into various abstract machines that generate architectural designs. The concept of effectors can be the unifying concept that allows the computational modelling of all architectural entities as active agents.

In the editorial to the issue of *Architectural Design* (Architectural Design, 2000) devoted to contemporary architectural processes, an emerging field is defined that optimises the state of the "in-between" as process and "systemic delay" as a major source of creativity. The concept of "in-between", as used in some of the projects featured in the issue, is based on the concept of "tweening" used in animation systems and not on the concept of poche'. The concept of poche' provides a different "in-between" paradigm. Systemic delay is defined as conceptual development in the time lag between an initial idea and its material form. This can also be related to the concept of poche', if the space-time of poche' (in its extreme characterisation, a poche' can be a space-time continuum), is considered a systemic delay between an idea and its realisation in material form.

As such, the concept of effectors and poche' can provide the means to mediate between idea and material form, between inside and outside, between performance criteria and space, in short, any condition that involves the mediation between two (or more) active principles. Though polyhedra are used in the examples in this paper, effectors can be used with completely curvilinear surfaces as well. Also, the polygons and polyhedra are not literal but represent networks of effectors that may constitute abstract machines.

4. REFERENCES

Architectural Design: Contemporary Processes in Architecture, 2000, Vol. 70, No. 3, Wiley-Academy, England, June, pp. 26-35 and pp.63-69.

Deleuze, G., 1988, *Bergsonism*, Translated by Hugh Tomlinson and Barbara Habberjam, Zone Books, New York, New York, pp. 107.

Deleuze, G. and F. Guattari, 1987, *A Thousand Plateaus: Capitalism and Schizophrenia*, Translated by Brian Massumi, University of Minnesota Press, Minneapolis, Minnesota, pp. 141 and pp. 510-514.

Mahalingam, G., 1998, "Representing architectural design using virtual computers," *Automation in Construction*, Vol. 8, Elsevier, New York, New York, pp. 25-36.

Mahalingam, G., 1996, "Object-oriented computer-aided design systems for the preliminary design of auditoria," *Journal of Architectural and Planning Research*, Vol. 13, No. 3, Locke, Chicago, Illinois, Autumn, pp. 214-229.

Venturi, Robert, 1966, *Complexity and Contradiction in Architecture*, Museum of Modern Art, New York, pp. 85.

Awareness Space in Distributed Social Networks

L.Gavin, S.Keuppers, C.Mottram & A.Penn
Bartlett School of Graduate Studies, University College London, UK

Key words: Algorithmic Form Generation, Distributed Workgroups, Space Syntax

Abstract: In the real work environment we are constantly aware of the presence and activity of others. We know when people are away from their desks, whether they are doing concentrated work, or whether they are available for interaction. We use this peripheral awareness of others to guide our interactions and social behaviour. However, when teams of workers are spatially separated we lose 'awareness' information and this severely inhibits interaction and information flow. The Theatre of Work (TOWER) aims to develop a virtual space to help create a sense of social awareness and presence to support distributed working. Presence, status and activity of other people are made visible in the theatre of work and allow one to build peripheral awareness of the current activity patterns of those who we do not share space with in reality. TOWER is developing a construction set to augment the workplace with synchronous as well as asynchronous awareness. Current, synchronous activity patterns and statuses are played out in a 3D virtual space through the use of symbolic acting. The environment itself however is automatically constructed on the basis of the organisation's information resources and is in effect an information space. Location of the symbolic actor in the environment can therefore represent the focus of that person's current activity. The environment itself evolves to reflect historic patterns of information use and exchange, and becomes an asynchronous representation of the past history of the organisation. A module that records specific episodes from the synchronous event cycle as a Docudrama forms an asynchronous information resource to give a history of team work and decision taking. The TOWER environment is displayed using a number of screen based and ambient display devices. Current status and activity events are supplied to the system using a range of sensors both in the real environment and in the information systems. The methodology has been established as a two-stage process. The 3D spatial environment will be automatically constructed or generated from some aspect of the pre-existing organisational structure or its information resources or usage patterns. The methodology must be extended to provide means for that structure to grow and evolve in the light of patterns of actual user behaviour in

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 615–628.
© 2001 *Kluwer Academic Publishers.*

the TOWER space. We have developed a generative algorithm that uses a cell aggregation process to transcribe the information space into a 3d space. In stage 2 that space was analysed using space syntax methods (Hillier & Hanson, 1984; Hillier 1996) to allow the properties of permeability and intelligibility to be measured, and then these fed back into the generative algorithm. Finally, these same measures have been used to evaluate the spatialised behaviour that users of the TOWER space show, and will used to feed this back into the evolution of the space. The stage of transcription from information structure to 3d space through a generative algorithm is critical since it is this stage that allows neighbourhood relations to be created that are not present in the original information structure. It is these relations that could be expected to help increase social density.

1. PROJECT FRAMEWORK

In a co-located team, members typically learn from a wide range of cues about the activities of the other members, about the progress in the common task and about subtle changes in group structures and the organisation of the shared task environment. Most of this group awareness is achieved without overhead effort. A distributed (virtual) team – even if its co-operation is based on state-of-the-art groupware systems – today has a far lower level of awareness and opportunity for spontaneous, informal communication. This reduces the effectiveness of the joint effort, and makes co-operation a less satisfying experience for the team members. The TOWER project aims to bring the wealth of clues and information that create awareness and cohesion in co-located teams to the world of virtual teams and to present them in a Theatre of Work.

TOWER supports group awareness and chance encounters through a 3D environment that is at the heart of the Theatre of Work. Avatars and their symbolic actions represent users and their current actions on shared objects while using a groupware application. Avatars of users who work in a similar context appear spatially close in the 3D environment. The integration of the Theatre of Work into a work setting opens new opportunities for spontaneous encounters and spontaneous contacts.

The project aims to create an awareness infrastructure that collects and manages sensor signals from the real work environments of the team members and events from the team's information space, and distributes them to trigger indicators appropriate for the team members;

1.1 Project

This paper will set out a methodology for the design of dynamic 3D worlds where architecture and objects represent the task-related information spaces of a team and their dynamics. Each work environment is essentially unique. Recognising this, we will develop a construction set approach to our system that recognises the signs of democratic communication 'pathmaking' to build a best-fit solution. This construction set enables an automatic creation and dynamic recreation of the virtual space based on a transformation of the elements of shared information spaces into elements and places of the Theatre of Work. A Space Module will be developed based on advanced spatial analysis techniques and an understanding of conceptual and physical distance to allow this construction set to build appropriate work environments.

The methodology has been established as a two-stage process. The 3D spatial environment will be automatically constructed or generated from some aspect of the pre-existing organisational structure or its information resources or usage patterns. The methodology must be extended to provide means for that structure to grow and evolve in the light of patterns of actual user behaviour in the TOWER space.

We have developed a generative algorithm that uses a cell aggregation process to transcribe the information space into a 3d space. In stage 2 that space was analysed using space syntax methods (Hillier & Hanson, 1984; Hillier 1996) to allow the properties of permeability and intelligibility to be measured, and then these fed back into the generative algorithm. Finally, these same measures have been used to evaluate the spatialised behaviour that users of the TOWER space show, and will used to feed this back into the evolution of the space.

The stage of transcription from information structure to 3d space through a generative algorithm is critical since it is this stage that allows neighbourhood relations to be created that are not present in the original information structure. It is these relations that could be expected to help increase social density.

In order to apply Space Syntax theory to the virtual world and its inhabitants, some specific data is required mapping the users, documents and their interactions. The methodology currently applied in 'real world' situations examines building layouts and maps the movement data, with this information the theory can be used to evaluating different strategies for changing the layouts in order to maximise any chosen interactions.

The principal data required for Space Syntax analysis is a comprehensive geometrical description of the environment and a log of the movements and interactions within it. In the application of Space Syntax to the Tower

environment, the situation is altered somewhat by the possibilities offered by virtual environments as regards changing, or morphing, geometries. The rules associated with movements and interactions can now be used to morph any elements (avatars, documents and geometries) within the environment in order to instigate any required interaction.

Each element has a series of attributes. All elements have attributes relating to their origin, destination and subject, some have 'length' (documents), time (avatars) etc. Each element, in an effort to represent itself, morphs its movements and geometries according to certain rules. The rules relating to visibility and adjacency determines an optimum position for the element within the world at any given time. The aim is to have the elements organise and re-organise themselves in a pattern using rules based on information from the communication log. Thus the world adapts and changes to reflect the preoccupations of the inhabitants and forms an environment within which location can take on meaning.

2. THE SPACE MODULE

The intention within the TOWER space is to build loosely on the way that social interaction and generation take place in the real environment. Here we summarise the main operative factors we need to take into consideration. First, the fundamental intention of TOWER is not to replace the real environment, but to supplement it. In particular it is to provide a means of increasing social density for organisations that of necessity are spatially fragmented and so sparsely distributed in real space. There are two main reasons why social density is important to the work organisation:

First, density makes possible communication and information flow. Low-density organizations face considerable overheads in achieving adequate information flows.

Second, density directly affects the efficiency with which organisations are able to generate new structures and divisions of labour in response to changes in their operating environment. Low-density organisations will tend to fossilise and find it difficult to adapt to change.

Already the trend in organisations is to even greater spatial fragmentation made possible by the growth in communication technologies. Although these technologies make it possible for organisations to function whilst more spatially dispersed than ever before, the effect of dispersion is negative overall. TOWER aims to help redress the balance by developing a new channel for creating spatial awareness and social density.

2.1 Social Density

In real environments we know that a substantial proportion of communications take place in a random and relatively unpredictable way. People working in a concentrated fashion are considered to be unavailable even if they sit in an open plan space, but as soon as they get up from their desk to go to an urgent pre-planned meeting they are regarded as available by those that they pass and are recruited into conversation. Our studies suggest that up to 80% of work related conversations in the office take place in this unplanned and essentially unplannable way. It is unplannable since you cannot predict that any individual will walk past at any particular time (Penn et al 1997). Now, consider that most work processes require people to construct certain patterns of movement between people, information sources and spatial resources in the work environment. In an organisation with a division of labour into different tasks and processes this means that different people will have different characteristic movement patterns and space use regimes. A single building will structure these movement patterns and certain spaces will be used in common between different individuals (everyone comes through the entrance in the morning, or uses the lift and stairs regularly, or the coffee machine, photocopier or toilets). However, just as importantly, building layout also separates groups from each other. Hospital layouts, for example, are very effective in separating doctor's movement patterns from those of patients except in the formal spaces where the medical interface is planned to take place, and quite a degree of sophistication is needed in hospital planning to achieve this. In corporate offices often clients and top management are separated from workers.

In social terms, the combination of work processes and the regular movement paths that those entail, with the spatial layout of the building, create a pattern of co-presence between different categories of building user (and co-absence between other categories). These patterns of co-presence are fundamental to the reproduction of the social form of organisations, since, by and large, you need to know people and who they are before you phone them. It is thus the people that you know since you see them regularly, even if you have never worked with them directly, that become a main resource for generation of new links in organisations. We call this field of co-presence of people 'social density'. Of course most organisations have a number of other methods of achieving density, including brainstorming, management training weeks, retreats and so forth, but all these are about spatialising groups of people that management think need to be put in touch. What built space can do is put people in touch that management could not have predicted needed to talk. What built space can also do is to reproduce only the existing organisational structure and eliminate other links – this is a

recipe for a fossilised organisation. The problem with dispersed organisations is that they tend towards fossilisation since active investment must be made in communication to overcome distance, and quite naturally, that investment is made on the rational basis of making existing links stronger, rather than supporting new links. This is where TOWER's main contribution will lie. By spatialising the organisation (albeit in a virtual awareness space) it will help to increase social density.

2.2 How do we achieve this?

The initial idea that we have been working with is that we need a two-stage process. We need to automatically construct or generate a 3D spatial environment from some aspect of the existing organisational structure or its information resources or usage patterns. We then need to provide means for that structure to grow and evolve in the light of patterns of actual user behaviour in the TOWER space. However, there is a risk that if we construct a spatial environment based directly on the information structure of the existing organisation, it will reproduce co-absence rather than generate new patterns of co-presence. A direct mapping of say the file hierarchy of the shared database would be expected to do this as file hierarchies are essentially treelike and to get between neighbouring branches one needs to pass through the trunk. We think that we need an indirect mapping therefore from information space to virtual space, and this mapping needs to generate virtual space with some quite distinctive spatial properties:

- it should be permeable everywhere (and in the main have few dead ends).
- it should be intelligible – that is have good local to global correlations.
- it should ideally both spatialise local groups and create larger scale relations between these groups.

An example of the spatial relations described in the last point could be viewed in terms of the distinctiveness of urban areas such as Covent Garden, Soho and Mayfair in London. You know when you are in each but cannot quite say where the boundary lies. In urban space this is achieved through the intelligibility structure of local to global relations.

The structure of the Space Module is based on a generative algorithm that uses a cell aggregation process to transcribe an information space into a 3D space. In stage 2 that space is analysed using space syntax methods to allow the properties of permeability and intelligibility to be measured, (the algorithm is explained in greater detail in later chapters and is referred to as 'culling') and then these are fed back into the generative algorithm. Finally, these same measures are used to evaluate the spatialised behaviours that TOWER users demonstrate in the space, and these are fed back into the evolution of the space .

The stage of transcription from information structure to 3D space through a generative algorithm is critical. It is this stage that allows neighbourhood relations to be created that are not present in the original information structure. It is these relations that could be expected to help increase social density. The feedback stage is also critical, since it is this feedback that will allows those neighbourhood relations that turn out to be useful to flourish and will eventually eliminate or replace those that turn out not to be useful.

3. CELL AGGREGATION

The primary functions of the Space Module are:

- to handle the generation of the 3D environment given an existing information infrastructure, and
- to handle the evolution of that environment based on patterns of use in the TOWER environment.

Both of these functions depend on the input information. An Event Notification Structure has been established in order to capture, distribute, and notify user operations or object state in both real and virtual work environments. The model is based on sensors, events, and indicators. Both sensors and indicators are part of the work environment. Sensors report user operations or object state encoded as events to the ENI server. The server accepts, collects, and processes these events. Indicators are used to visualise events. There are two modes for indicators to get notified about events: in the first mode the indicator ask for events and in the second mode the indicator is notified of events by the server.

The description of the existing information infrastructure is derived from logs of actual usage of information sources. By tracking time-based patterns in that usage, inferences are drawn on likely relevance of different items of information. These are used to give shape to the spatial relations between information resources as represented in the TOWER world. In this sense the environment will grow according to actual usage patterns of the underlying information resources.

The main paradigms that are being used are aggregate models of environment construction. In these, as each information resource appears in the ENI event database a spatial representation is created within the TOWER world. In its simplest form a special agent navigates the entire information space and creates special ENI events at each node so that the event database comprises a full description of the whole information space. Simple sets of rules govern the location and form of the representation and the way that it is related spatially to other information resources. These rulesets implicitly

define the structure of space left over in the environment between each of the information resources. The precise form of the aggregation ruleset is therefore critical to the definition not only of local relations between different resources, but also of the global spatial configuration of the whole environment. However, both local relations and global configuration are not directly coded into the ruleset, but emerge dynamically as a result of the application of the ruleset to a specific log of information usage. Given a different usage pattern, a differently configured environment would emerge.

There are two main functional requirements of the rulesets. First, it is important that the local relations between information resources be 'sensible'. This means that there should be some relatively obvious logic to the local disposition of resources with respect to each other. Second, the whole configuration of the resulting aggregation should be relatively intelligible. It is our expectation that by aggregating according to an actual history of usage the local relationships will emerge as an aspect of the process. Thus information resources that are used in temporal proximity in an actual log will be located in spatial proximity in the TOWER environment. Our definition of 'intelligibility' of a spatial configuration is based on the space syntax concept of the prediction of global position from local information. Aggregation processes are being developed that lead naturally to relatively intelligible configurations.

The simplest of these aggregation processes is the 'beady ring', settlement form identified by Hillier & Hanson (1984). In this aggregation a closed cell open space dyad forms the aggregation unit. The rule is that open space must attach to open space. The first dyad is located and oriented at random. The next can take one of three open faces to attach its open space, and each of those sites can be connected in three possible orientations. Again each of these is selected at random. The process continues until all information resources have been located. The problem with this form of aggregation is that although relatively small aggregations are syntactically intelligible, as aggregations grow the relationship between local and global spatial properties breaks down.

3.1 Culling

For this reason a further stage of 'information culling' has been included in the process. In this stage, the environment is reviewed in terms of the intervisibility of different information resources. Information resources are rated according to measures of similarity or relevance, and these culling procedures seek to maximise the intervisibility of those resources that are most strongly related. This process tends to lead to a relocation of certain

resources within the environment, and to the removal of resources from locations where they block the intervisibility of other resource pairs.

4. RULESET

4.1 Identifier and Descriptor Attributes

Each object in the world has two sets of attributes. The Identifier Attributes are automatically set by ENI events. They can include any 'field' in the ENI event list e.g. 'creator', 'reader', 'length of document', keywords. The second set of attributes are the Descriptor Attributes. These could take the form of the settings for descriptors such as the geometric form of the representation (these are currently cubic 'building blocks' in the early prototype implementation, but could be cylindrical 'trees' or any other geometric representation), colour, height, transparency, hue, and text labels.

Decisions have to be made regarding the mapping of identifier to descriptor attributes. Currently the early prototype world reflects the frequency of people accessing a document (summed from 'creator', 'reader', 'modifier' and 'date' fields in the ENI event record) represented by the height of the building block. There are, however, a number of other event attributes that we might choose to represent, such as MIME type, document size, creator, modified date etc. There are also a number of representation factors other than height, including change of geometry, colour, transparency, textual labelling or a combination of these. Location can also be considered a representational factor, however this is treated separately in the system since it is qualitatively different in that it allows the representation of the relationship between different events or information resources. The Attribute Matching System handles location.

4.2 Attribute Matching System

The Attribute Matching System is used to compare existing objects with a new object in order to decide where the new object should be placed within the world. Currently the early prototype matching system allocates points purely on the basis of similarity of Identifier Attributes. This assessment is carried out for each new object against each existing object in a 'snap' (card matching) type game e.g. the new document was 'created by Smythe' and the existing document was 'created by Smythe' would gain the match 5 points. If both documents were 'read by Prinz' the match would gain another 5 points. Currently if there is a free space next to the existing document this gains an

additional point. The new object is eventually placed in a position next to an existing object with the highest match points and a free space.

The points allocation structure in the Attributes Matching System is an area in which we anticipate further development taking place. Users may wish to prioritise certain attributes and or give others negative points. The main issue here is to develop methods for location of information resources or event representations in which locations become meaningful to the user. This will inevitably differ from one end user application to another, and so the approach adopted at this stage is to develop a generic method for determining location based on attribute matches, where the specific attribute values can be user defined.

4.3 Ideal Neighbours and Actual Neighbours

Ideal Neighbours are the objects that score the highest points through the Attribute Matching System. Actual Neighbours are the objects that score the highest points through the system and have a free space and so are chosen as neighbours. Ideally, the Ideal Neighbour will be the Actual neighbour, however as a TOWER world grows free spaces may get used up and less than ideal locations for new blocks will be used. These definitions are utilised in the 'Culling Routine' in order to re-arrange the world through best Ideal Neighbours and sight lines.

4.4 Placement Rules

Once the Actual Neighbour has been identified the placement rules are followed. Each object is currently represented in the early prototype as a 'building-block' of enclosed (private) space with an adjoining 'plot' of open (public) space. The placement rules require neighbouring open plots to meet as in Figure 1. The cumulative effect of these placement rules creates blocks of private spaces, and streets and squares of public spaces. The effect is to generate a continuously accessible pattern of open space, irrespective of the nature or sequence of events and attribute matches describing the information space. It is impossible to generate areas of inaccessible open space.

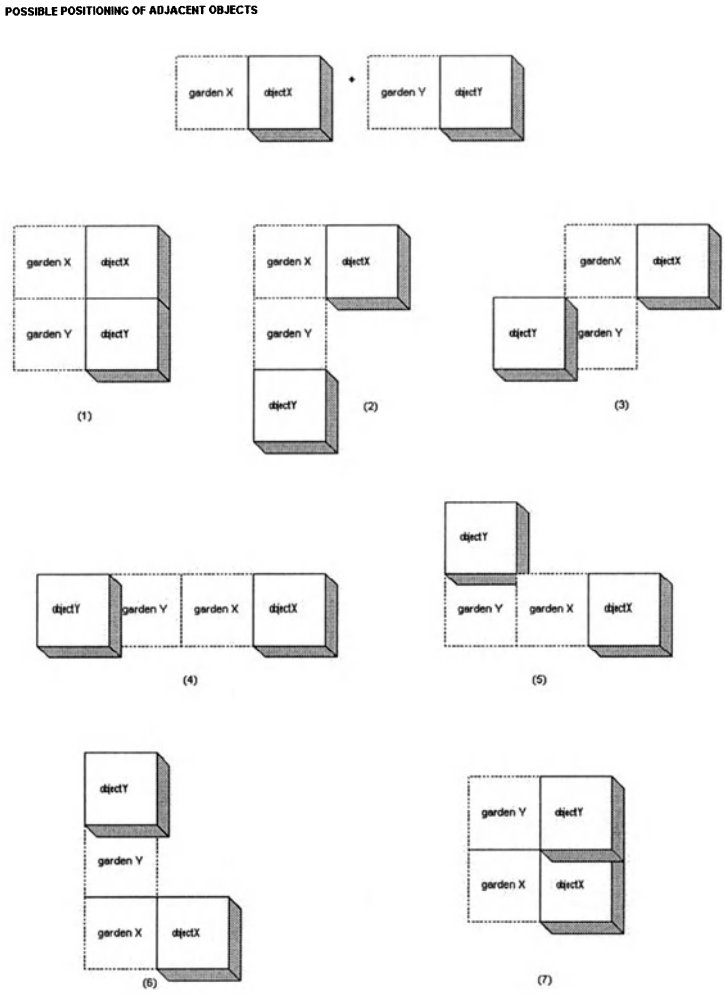


Figure 1. Placement Rules

The resulting patterns of development can be shown to have quite specific proto-urban qualities. All areas of open space relate directly to building blocks and as a consequence of building blocks related by attribute matching are likely to be visible to one another. As the space pattern grows above a relatively small size, local and global spatial properties are found to become correlated and this gives the spatial pattern a natural property of intelligibility. As the aggregation grows still further, however, intelligibility diminishes, and this calls for further processes to ensure that the world remains scalable. The main means of achieving this is the culling process.

This not only ensures that related items of information are inter-visible, but also serves to increase the degree of spatial interconnections between different parts of the evolving aggregation, and ensures increased intelligibility as a system grows larger.

4.5 Life Points

The Life points system helps to reflect a documents history within the world. Life points are added whenever a document is accessed, but subtracted whenever a 'culling' takes place. Presuming that the culling will take place on a regular basis, say daily, the document would eventually disappear if it was not accessed after a certain number of days. This process also allows for a degree of change in the layout over time.

5. RESULTS

5.1 Examples of the World Building and Culling Rulesets

Figure 2 shows the growth and culling process for a TOWER workspace using dummy event data. In each stage of the process new building blocks are added to the world representing new nodes in the database hosting the collaborative workspace. These are coloured according to context attribute values denoting degrees of similarity between items of information in the information space. The process of aggregation tries to bring information items that are closely related together in space, however as the world grows it becomes increasingly difficult to satisfy this requirement without rearranging the location of existing blocks. This is when the culling process comes into force. As time progresses the number of new information items reduces, and events increasingly involve hits on existing information items in the world. These events are represented in the prototype world as increases in the height of a block. Thus the most frequently visited information items become TOWERs in the landscape. As the world develops, the rules lead to a degree of structuring of the landscape, but not to a precise correspondence between layout and colour. This is a result of the fuzzyness of the context attributes and the fact that there are numerous competing associations between information items. The process leads to associations in space between items of information that were not adjacent in the original hierarchical file structure in the database. This is the key aspect

of the TOWER world design that will lead to users being brought into contact with new and relevant items of information .

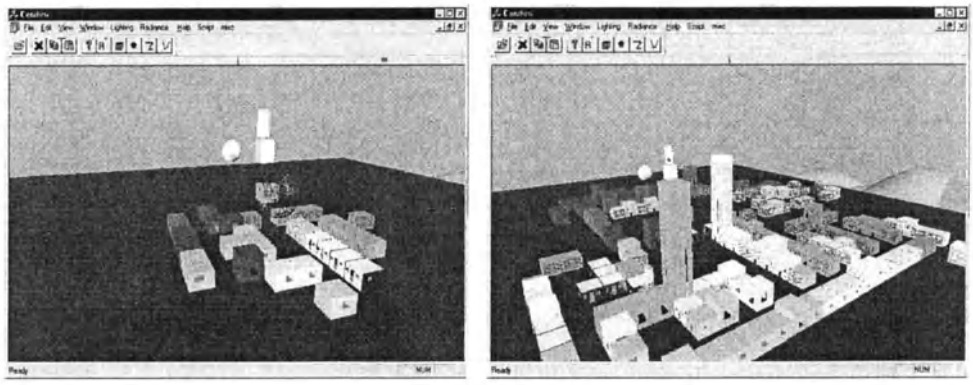


Figure 2. World Growth

5.2 The resulting spatial layout

The aggregation ruleset used to locate 'building blocks' in the TOWER world leads to a pattern of connections between open space in the world. This pattern is a characteristic of the rules used to aggregate and to relocate blocks during culling. Tests have been carried out using Space Syntax analysis techniques to evaluate the resulting spatial layout of the TOWER world. The intention is that as the information space grows it should maintain a degree of intelligibility. In space syntax analysis it is possible to provide a metric for the degree of intelligibility of a spatial system in terms of the degree of correlation between local and global measures of the graph describing intervisibility of points in space. Figure 3 shows measures of local (on the left) and global (on the right) measures for the TOWER world as it grows. Although there is a lack of correlation at the start of the process, as aggregation and culling proceed the result is that local and global measures become increasingly correlated. The final result of culling is to remove blocks in the centre in order to provide intervisibility between related information items.

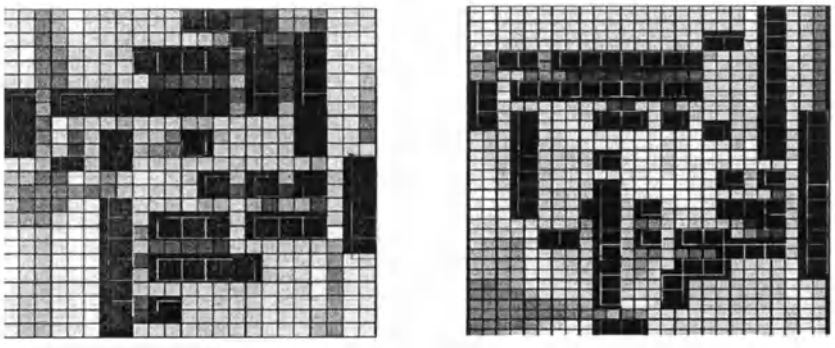


Figure 3. Connectivity Patterns

6. REFERENCES

Hillier, W. and J. Hanson, 1984, *The Social Logic of Space*, CUP, Cambridge.

Penn, A., J. Desyllas, J. and L. Vaughan, 1999, The Space of Innovation: Interactionand Communication in the Work Environment, in *Environment and Planning B:Planning and Design*, (1999), Vol. 26, 193-218, Pion, London.

Bates, Marcia J., 1996, "Document Familiarity in Relation to Relevance, Information Retrieval Theory, and Bradford's Law: The Getty Online Searching Project Report No. 5." in *Information Processing & Management* 32 (1996): 697-707.

Belisle C., R. Zeiliger, T. Cerrato, 1997, *Integrated Cognitive Engineering at the Interface: A Tool Mediation Perspective*, http://www.ish-lyon.cnrs.fr/labo/LIRE/CT97.htm.

Rose, A., W. Ding, G. Marchionini, J. Beale, and V. Nolet, 1997, *Building an Electronic Learning Community: From Design to Implementation*, http://www.cs.umd.edu/TRs/authors/Gary_Marchionini.html.

Thatcher, A., *Determining interests and motives in WWW navigation*, Psychology Department, University of the Witwatersrand, WITS, 2050, South Africa, http://cyberg.curtin.edu.au/members/papers/66.shtml

Evaluation of Design Performance through Regional Environmental Simulation

Robert Ries and Ardeshir Mahdavi[1]
University of Pittsburgh, Carnegie Mellon University[1]

Key words: Environmental simulation, design decision support, life cycle analysis

Abstract: Computational building simulation tools have historically viewed buildings as artefacts isolated and disconnected from their contexts. At most, the external environmental conditions have been viewed as outside influences or stressors encapsulated in, for example, weather files for energy simulation or sky models for lighting simulation. In the field of environmental assessment, life cycle analysis (LCA) has followed a similar path of isolating the artefact under analysis from its context. Modeling the building artefact as a participant in multiple contexts over time so that the interactions and dependencies between the regions and the building can be adequately explored in the design process requires support for the modeling of regional areas, as well as the artefact and the related life cycle processes. Using computational design and evaluation tools can provide the computing capability required for effective design decision support. This paper presents the implementation of the affordance impact assessment method and the regional environmental simulation in Ecologue. Ecologue is the computational tool for life cycle environmental impact assessment in the SEMPER integrated building design and simulation system. Ecologue contains a building model and an environmental model. The building model is automatically derived from the shared building model of the SEMPER system. The environmental model is a combination of a representation of the processes and emissions occurring in the life cycle of buildings and an impact assessment model. The impact assessment model is a combination of a context model of the physical characteristics of a region and a sub-regional fate and transport model based on the fugacity concept.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 629–642.
© 2001 *Kluwer Academic Publishers.*

1. INTRODUCTION

1.1 Buildings interaction with the environment

Buildings exchange mass and energy flows with their surroundings and the wider environment in multiple ways throughout their service life. A life cycle framework is essential to understanding and evaluating these interactions. A life cycle assessment is typically broken down into six stages: raw materials acquisition, manufacturing, processing, transportation, use/reuse/maintenance, and recycling/waste management. An equivalent set of stages for a building or building component would be raw material mining or harvesting (i.e., mining clay, stone, ore, gravel, or harvesting wood), building material production (i.e., milling and fabrication, metal smelting, rolling, and shaping, and assembling), building construction (site work, structural frame, enclosure, and finish work), building operation (heating and cooling, lighting), and building decommissioning (reuse, recycle, disposal).

The principal raw material acquisitions for buildings fall into three categories: abundant materials, such as sand, clay, and gravel; potentially renewable resources, such as wood and fibre; limited and non-renewable but possibly recyclable resources such as metal ores (bauxite, iron and copper ores), and plastics from fossil feed stocks. Impacts from raw material acquisition can include land degradation, hazardous waste generation from extraction and refining, and resource depletion.

The building component production phase includes the manufacturing of constituent materials, such as metals and rough lumber through the fabrication of products such as studs, cement, windows, and doors. Materials and components can be highly finished in a factory or minimally processed with additional work required on site. Typical impacts from this stage are releases related to the manufacturing processes and energy use in material production and fabrication.

The principal impacts in the building construction stage are waste generation and energy and equipment use for assembly. The construction waste generated by the construction of an average home in North America was found to be approximately 20 kg • m^{-2} of floor area 4,000 kg for an average 185 m^2 (2,000 ft^2) house (NAHB 1996 see *Figure 1*).

The principal impacts in the building operation phase are energy-related (heating, cooling, lighting and equipment), maintenance-related (repair and replacement of building components) and related to indoor environmental quality. Despite the introduction of energy efficiency measures, residential and commercial energy related greenhouse gas emissions have outpaced the rate of population increase (*Figure 2*). Operational loads can comprise a significant proportion of the total life cycle environmental loading (*Figure*

3). Additionally, indoor environmental quality is an important occupant health issue. Concentrations of pollutants can be significantly higher indoors when compared to ambient air (Ott and Roberts 1998).

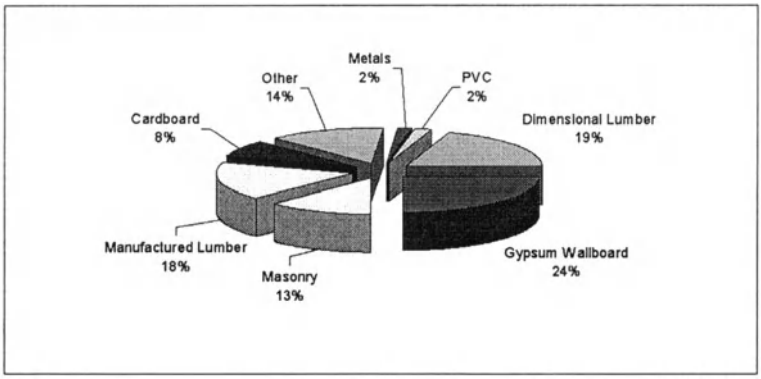


Figure 1. Characterization of the construction waste generated by the average home construction in the United States (NAHB, 1996)

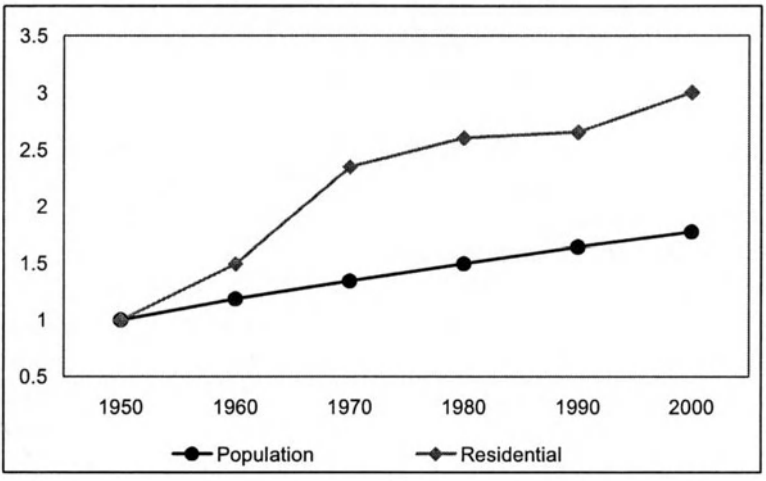


Figure 2. Increase in energy-related greenhouse gas production in the US 1950-1998 relative to population in the residential building sector (1950 = 1)

Building decommissioning and demolition generates primarily inert materials that have historically been landfilled. Principal components of demolition waste are wood and concrete, which together comprise two-thirds of the demolition waste of an average house. The most common materials recycled are concrete and asphalt, wood, asphalt shingles, metals,

and drywall. Five states report an average of 48% recycling rate for construction and demolition debris (USEPA, 1998).

Considering the implications of the life cycle of buildings to the understanding of its environmental impact, a life cycle assessment framework should be considered an essential part of the design of the built environment.

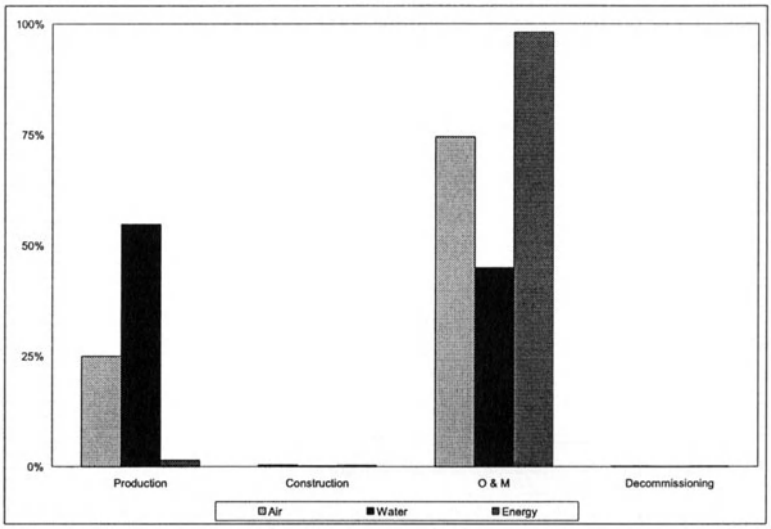


Figure 3. Distribution of life cycle air, water, and energy use impacts among four life cycle stages (Ries, 2000)

1.2 Life cycle assessment

LCA is defined as a four phase process (Fava et al., 1991) consisting of goal definition and scoping, which defines the objectives of the study and determines the analysis boundary; inventory analysis; impact analysis; and improvement analysis, which is an evaluation of the environmental loads identified in the previous stages in order to determine modifications to the product or process that will reduce environmental impact. Within the LCA framework, environmental assessment methods differ primarily in how the inventory analysis is performed and in how the impacts are evaluated.

Life cycle inventories are most commonly calculated as the sum of pollutant releases and/or energy use for the processes related to the material or component in question. Life cycle impact analysis evaluates the effects of the constituents of the life cycle inventory. Estimating the effects of the life cycle inventory is perhaps the most difficult stage of the LCA. The methods developed have used a number of different strategies. The most

straightforward and simple methods use factors such as energy use or the mass of pollutant emissions from the life cycle inventory as indicators of environmental performance. Other methods use categorization and weighting strategies. These gauge the effects of the emissions and use a weighting or effect formulation to normalize, compare, and group emissions so that a single indicator value or multiple values can be calculated.

Impact analysis methods have evolved from the energy- and pollutant mass-only methods. However, a number of limitations remain. In current LCA practice, impact analysis is based on the life cycle inventory stage, which is a mass- or intensity-only accounting of pollutant emissions and material and energy use. The aggregation of the life cycle releases into a single dimension has the following constraints. This type of aggregation does not take into consideration the varying intensities (e.g., emission or use per unit time) that occur in actuality. A short-term high intensity emission release may aggregate to the same mass as a long-term low intensity release, although the environmental impact may not be equivalent. Aggregation also does not take into account the spatial distribution of an emission release. Emission releases equivalent in mass terms could be distributed over different volumes of media, result in different concentrations and therefore potentially different environmental impact. Emission inventories also do not consider the characteristics of the context or region where releases occur. As a result, the sensitivity of the context to an emission release cannot be included in the analysis. Additionally, most LCA impact assessment methods require a comparison of alternatives, but do not necessarily determine whether any of them are within the ability of the ecosystem to sustain the environmental loading over a period.

Analysis of the results of a life cycle analysis of a prototype residential home using the Critical Volume method of environmental impact assessment illustrates some of these points (Ries, 2000, Etterlin et al., 1992). *Figure 3* shows that relatively, the building material production and operation and maintenance phases are the principle life cycle phases for environmental impact. In addition, when aggregated over the life of the building, estimated at fifty years in this case, the majority of the impacts from emissions to the air and energy use occur in the operation phase. *Figure 4* shows the relative emission and energy use rates for the same prototype residential building. The emission impact and energy use rate is defined here as the unit impact or use over unit time. When comparing the two figures, one can see that the rate of environmental impact from emissions and energy use are greatest in the building material production phase, and the relative importance of the operation phase has been reduced.

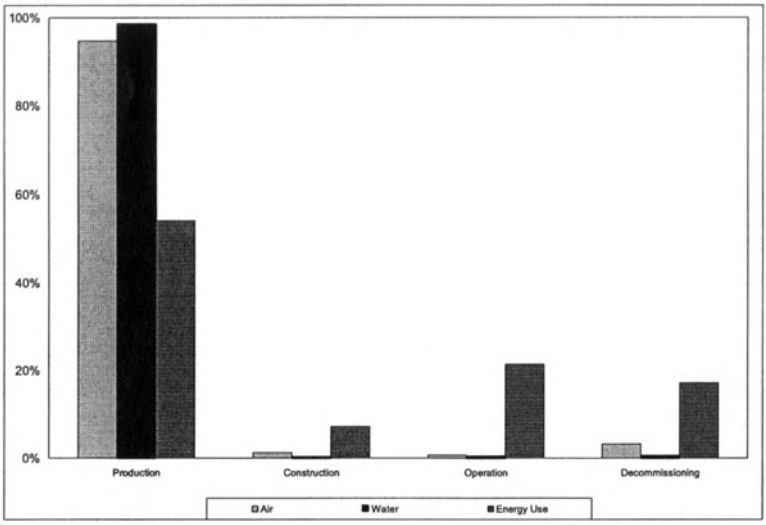


Figure 4. Distribution of air, water, and energy use impact rates among four life cycle stages

These limitations have motivated the work toward the impact assessment method called affordance. Affordance is an indicator that calculates a spatial allocation for emissions and resource usage and has units expressed in kg · m^{-2} · yr^{-1}. The implementation of affordance described here uses a model evaluative environment, which allows for the consideration of the emission rate and the spatial distribution of an emission release in the indicator calculation. The evaluative environment models the characteristics of the context or region where releases occur. As a result, the sensitivity of the context to an emission release can be included in the indicator calculation. In addition, the affordance concept is similar to carrying capacity in that it evaluates the ability of the ecosystem to sustain the emission rates over a period. Lastly, the allocation of an emission rate per unit area makes the indicator useful for land use and resource planning. The context-based representation of a life cycle inventory with a model that includes spatial and temporal distributions of environmental emissions will improve environmental evaluation of the design, construction and operation of the built environment and lead to improved design decision-making.

2. IMPLEMENTATION

The majority of current LCA software tools are general purpose and stand-alone, i.e., outside of a CAD environment. A number of software tools have been developed specifically for the building industry. Examples are

ATHENA (Trusty et al., 1998), LEGOE (Hermann et al. 1998), and BEES (Lippiatt, 2000). ATHENA and BEES are stand-alone, and LEGOE is an example of an integrated software tool that includes energy analysis and LCA.

In the stand-alone systems, the user specifies the construction system and building components through an interface that describes the constituent elements of the building, such as walls, floors, foundations, beams, or columns. The user specifies the area or quantity of each building element. Once the building description is complete, the tool applies the environmental factors for the specified components and finishes. The environmental factors are based on detailed LCA data, but this is not accessible to the user. Therefore, the capability to construct custom assemblies out of materials or components in the tools is limited. In-depth analysis of the causes of the resulting environmental impacts is limited because the underlying LCA data is not available. These systems also require at least two materials, components, or building descriptions for comparison.

The affordance impact assessment method has been implemented in Ecologue (Ries 1999). Ecologue is the computational tool for life cycle environmental impact assessment in the SEMPER integrated building design and simulation system (Mahdavi, 1999). Ecologue contains a building model and an environmental model. The building model is automatically derived from the shared building model of the SEMPER system. As a result, the environmental assessment tool does not require separate user input of the building description. A portion of the building description includes references to, for example, technical elements and construction types. These additional elements are fully described from domain viewpoints in database representations that are accessed by Ecologue. The combination of the building model and the database elements are used by Ecologue to generate the building representation from the environmental domain viewpoint. An interface is provided to create new and edit existing domain-specific database representations. The environmental portion of the Ecologue model is a combination of a representation of the processes and emissions occurring in the life cycle of buildings and an impact assessment model. The impact assessment model for affordance is a combination of a context model of the physical characteristics of a region and a sub-regional fate and transport model based on the fugacity concept (Mackay, 1991).

2.1 Affordance Impact Assessment Model

Determining the environmental impact value of a process in a context using the affordance method requires the modeling of the fate and transport

of the process-related emissions, and the estimation of the resulting concentrations in the context. The regional or context model in Ecologue is a multi-compartment model (see *Figure 5*). A compartment is defined by its physical characteristics, such as the size (area and volume) and the rates of advective flow of the media. A fugacity-based model is used within each compartment. Fugacity is a property of a substance that is used for predicting mass and concentration distributions, reaction characteristics, and persistence of a chemical released into an environment.

The fate and transport processes modeled are emissions to air, soil, and water (E_i), transmissions between media (D_{ij} representing diffusion, deposition, and runoff), reactions (R_i), and advection (A_i). Mass balance equations are used to determine the distribution tendency and concentration. The evaluative environment is not intended to simulate the real environment, but is intended to provide the behavioral characteristics of the substance in terms of partitioning among the media (air, water, soil, and sediment).

The context is modeled as a bounded system, and therefore no transfers of mass occur across the boundary. Conceptually, the scale of the context model can range from a world model, a geographical region, or rooms in a building interior. A regional context model can correspond to a naturally defined area, such as a watershed.

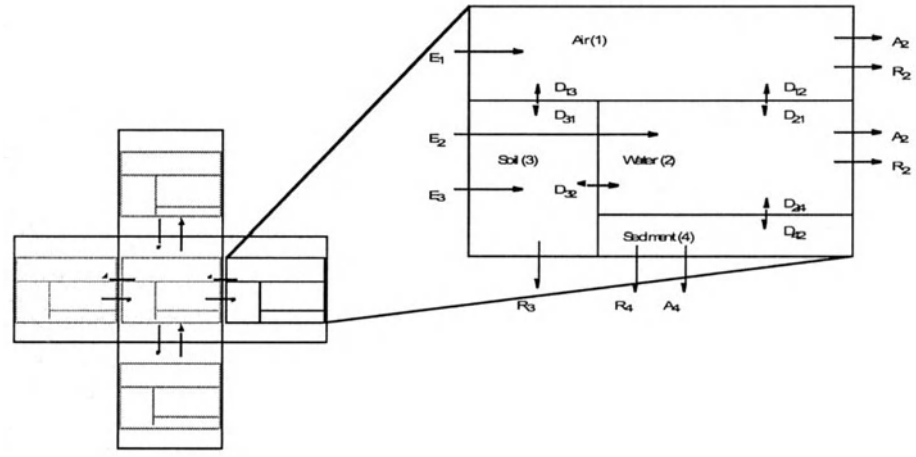


Figure 5. A five-compartment system with mass transfer between compartments and an expanded view of a compartment showing the transfer phenomenon modeled in the evaluative environment

Process and Emission Model

Processes and the related components (see *Figure 6*) are the principal elements used in the calculation procedure. Processes model activities in the

environment, and are related to an element of the Ecologue building model. Each process can be composed of multiple processes, each with a set of related emissions and one related context (see *Figure 7*). Emissions are modeled as a set of chemical and physical attributes. The interrelationship of the characteristics of the emission and the context together determine the distribution and concentrations of the emission in that context.

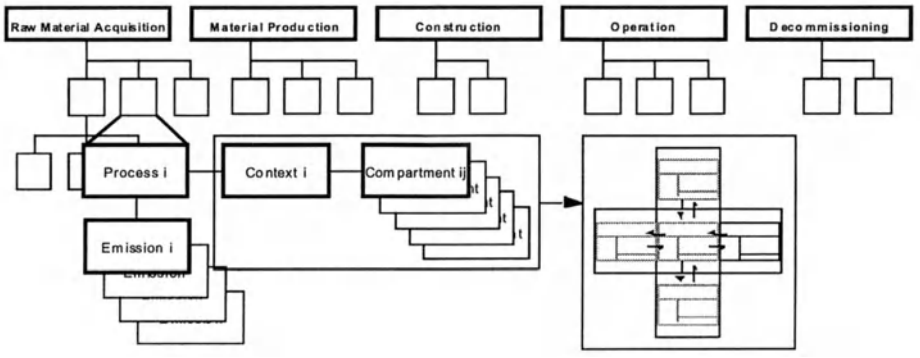


Figure 6. Relationship of the process model to the building life cycle

2.2 Calculation Procedure

In the affordance method, the allowable emission rate for each emission in each process in the context is calculated using the model described in the previous section. The calculation procedure evaluates each emission-context pair. The combination of the context characteristics, including the transfer between compartments, the emission properties, and the emission rates determines the allowable concentrations. The model is calculated iteratively and the allowable emission rate is found when the concentration in one of the media in a compartment reaches the target concentration range (Sittig, 1994), within a tolerance factor. The allowable emission rate is then divided by the developable area in the process compartment, resulting in the allocation of the emission rate per unit area.

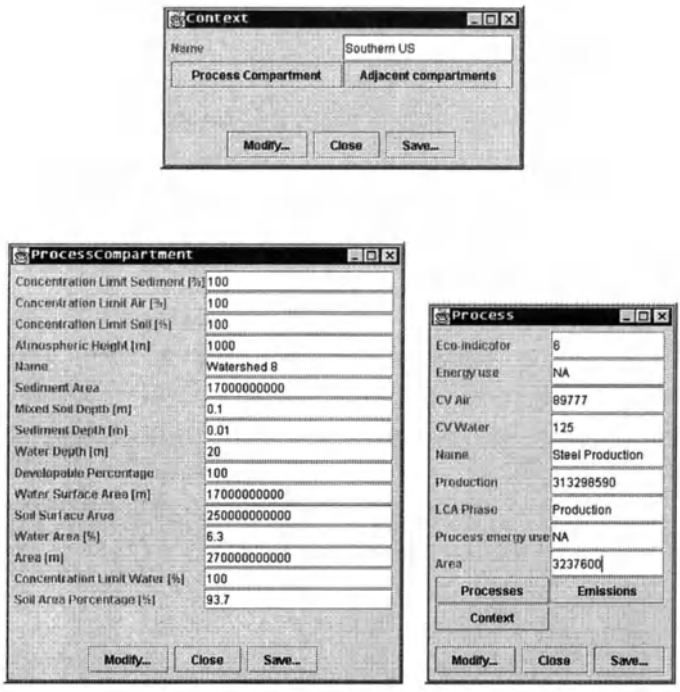


Figure 7. User interface windows for the specification of a context, compartment, and process

3. ILLUSTRATIVE EXAMPLE

3.1 Emission and Context Characteristics

This scenario presents the affordance calculations for a process in the material production portion of the building life cycle. The process is a typical manufacturing process with four emission types: formaldehyde, methanol, phenol, and methyl ethyl ketone. All emissions are released to the air. The process occurs in a facility with a 4,000,000 m^2 land area.

For calculating the allowable emission rates, the process is modeled in a five-compartment context of the type described in the previous sections and shown in *Figure 5*. Each context has the same land and water areas. The concentration limits for each emission and each media are set at 100% of the legal threshold limit for all compartments. The assumption is that one half of the context area will be developed, with the remainder undeveloped.

3.2 Results

The results of the affordance calculation are presented in *Table 1*. The calculations show that the spatial emission rate is exceeded largely for two of the emissions - formaldehyde and phenol, and to a lesser extent by methyl ethyl ketone. Methanol is below the allowable rate, resulting in an impact less than 1.

Table 1. Results of the illustrative affordance calculation

Emission	Actual spatial emission rate e $[kg \cdot m^{-2} \cdot yr^{-1}]$	Calculated affordance emission rate e_{max} $[kg \cdot m^{-2} \cdot yr^{-1}]$	Impact $(e \cdot e_{max}^{-1})$
Formaldehyde	$4.30 \cdot 10^{-4}$	$4.91 \cdot 10^{-8}$	8790
Methanol	$8.80 \cdot 10^{-5}$	$8.62 \cdot 10^{-4}$	0.10
Phenol	$2.80 \cdot 10^{-4}$	$4.02 \cdot 10^{-8}$	7070
Methyl ethyl ketone	$1.69 \cdot 10^{-5}$	$1.17 \cdot 10^{-6}$	14
		Total impact:	5640

The results indicate that if all developable areas of the evaluative context were to release substances at the rate that this process does, it would cause a concentration in excess of the legal threshold limit for four of the five substances. Therefore, the emission rates of this process are above the capacity of the ecosystem, and should be reduced. The other alternative would be to reduce or eliminate the rate of emission of these substances on additional areas of the region. Similar to emission credits, this could conceivably be achieved through negotiation.

4. CONCLUSIONS AND FUTURE WORK

A first step to enhancing the fugacity-based context model would be to expand the scale of the model. Coupling multiple context models by linking the input and output mass flows of adjacent multi- compartment context models would allow a larger-scale regions to be modeled iteratively (see *Figure 8*).

Linking the input and output mass flows of context models would allow the development of a spatial hierarchy of nested context models. The models within a hierarchy multiple scales would be used to assess impacts that occur at dissimilar spatial scales. Each of the multi-compartment context models could be scaled to the appropriate size for each assessment desired.

The affordance method could be realized with alternative or additional environmental and geophysical models that would utilize a model to calculate affordance values for contexts depending on the type of the mass or resource flow. A limitation of the current fugacity-based fate and transport model is that it was developed for organic compounds, and is not universally applicable for all types of emissions. Incorporating multiple models would improve the scope of the current application.

Assessing the environmental impact of a large-scale artefact with a long service life, which is the product of a heterogeneous system of processes, such as a building, is a complex problem requiring a large amount of information. Additionally, there is a significant amount of uncertainty in the analysis. Uncertainty is introduced into the analysis in three principal areas: in the estimation of the quantity of emissions from processes in a life cycle inventory; in the estimation of the environmental impact of those processes and activities in a life cycle impact assessment; and in the prediction of future building activities and functions. A life cycle analysis with multiple stages can propagate uncertainty throughout the model, leading to a result with a wide range of variability. Incorporating the calculation of uncertainty into the data model and assessment calculation would improve the current implementation.

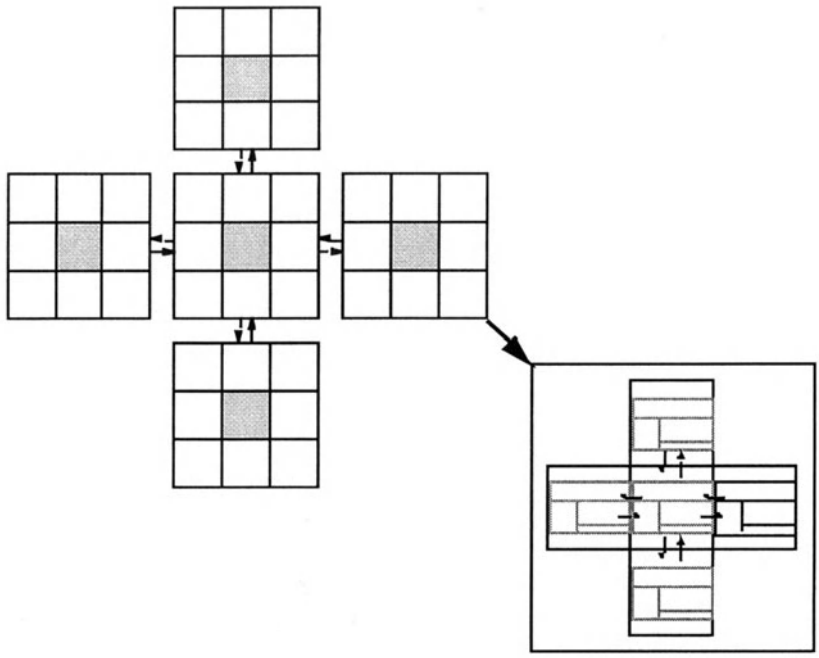


Figure 8. Expansion of the multi-compartment context model into multiple coupled contexts

LCA has provided a framework for the investigation of the environmental impacts of products and processes, and has led to the development of various methods and software tools developed for or applied to the building industry. Impact analysis methods have been consistently improved since the initial energy and pollutant mass methods. The affordance impact assessment method presented here has the potential to address some of the limitations of the mass- or intensity-only accounting of pollutant emissions and resource use used in most methods with a spatial allocation formulation for environmental impact. The affordance method allows for the consideration of the emission rate and the spatial distribution of an emission release in the indicator calculation. The modeling of the characteristics of the context or region where impacts occur allows the carrying capacity and sensitivity of the ecosystem to be included in the indicator and makes the indicator useful for land use and resource planning.

5. REFERENCES

Etterlin, G., P. Hursch, M. Topf, 1992, *Ökobilanz, Ein Leitfaden für die Praxis*. B.I. Wissenschaftsverlag, Mannheim Germany (in German).

Fava, J. A., R. Denison, B. Jones, M. A. Curran, B. Vigon, S. Selke, J. Barnum, 1991, *A Technical Framework for Life Cycle Assessment.* Society for Environmental Toxicology and Chemistry, Pensacola, FL.

Hermann M, Kohler N, König H, Lützkendorf T, 1998, "CAAD System with Integrated Quantity Surveying, Energy Calculation, and LCA", Proceedings: Green Building Challenge '98, Vancouver, Canada. V. 2, 68-75

Lippiatt B C, 2000, BEES 2.0: Building for Environmental and Economic Sustainability Technical Manual and User Guide. NISTIR 6520, National Institute of Standards and Technology, Gaithersburg, MD.

Mackay, D., 1991, *Multimedia Environmental Models: The Fugacity Approach.* Lewis Publishers, Chelsea, MI.

Mahdavi, A., 1999, "A comprehensive computational environment for performance based reasoning in building design and evaluation", *Automation in Construction 8*, p. 427 - 435.

NAHB, 1996, *Residential Construction Waste: from Disposal to Management.* National Association of Home Builders Research Center, Inc. Upper Marlboro, MD.

Ott, W. R. and J. W. Roberts, 1998, "Everyday Exposure to Toxic Pollutants." *Scientific American*, vol. 278, no. 2.

Ries, R., 2000, "Environmental life cycle assessment of continuous customization: A case study of a residential building". *Continuous Customization in Housing, Proceedings of CIB 104 Open Building Implementation, Tokyo, Japan,* CIB Report Publication 254, p. 197-206.

Ries, R., 1999, "Computational Analysis of the Environmental Impact of Building Designs." PhD thesis, Carnegie Mellon University, Pittsburgh, PA.

Sittig, M., 1994, *World-wide Limits for Toxic and Hazardous Chemicals in Air, Water and Soil.* Noyes Publications, Park Ridge, NJ.

Trusty W B, Meil J K, Norris G A, 1998, "ATHENA: A LCA Decision Support Tool for the Building Community", Proceedings: Green Building Challenge '98, Vancouver, Canada. Vol. 1, 39 - 46

USEPA, 1998, *Characterization of Building-related Construction and Demolition Debris in the United States.* EPA530-R-98-010, U.S. Environmental Protection Agency, Office of Solid Waste, Municipal and Industrial Solid Waste Division, Franklin Associates, Ltd. and TechLaw, Inc.

Non-linear Structural Analysis as Real-Time Animation

Borrowing from the Arcade

Kirk Martini
University of Virginia

Key words: Structural analysis, interactive design, animation

Abstract: The paper describes a computational method commonly used in interactive computer graphics and games, and demonstrates its application to structural engineering problems, using a prototype program called *Arcade*. The method enables a new model of interaction in structural analysis, where the simulated structure responds to user input in real time, in the same way that computer games respond. The method shows good engineering accuracy in simple verification problems involving the non-linear phenomena of buckling and beam yielding. The method offers the potential to make non-linear, time-history analysis a much more common method in engineering practice, and to bring a greater emphasis on non-linear, dynamic structural behaviour in structural education.

1. OVERVIEW

One of the significant recent trends in interactive computer graphics is the incorporation of physics and elasticity, particularly in games (Heckler 1996). A common approach is to model objects as a system of particle masses connected by elastic springs, commonly known as *particle systems* (Witkin and Baraff 1997b). Particle systems provide a simple means to model deformable objects which exhibit visually realistic physics at interactive frame rates on conventional personal computers.

Particle systems are related to computer-based structural analysis methods used in the design of buildings, where the springs of the particle system correspond to finite elements used to model structural members. Although both methods can be used to analyse structures, the particle system

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 643–656.
© 2001 *Kluwer Academic Publishers.*

approach has received little attention in building structural analysis, primarily because the particle approach is computationally less efficient by orders of magnitude, and is more prone to numeric instability.

However as computation speed and capacity continue to increase, the computational shortcomings of the particle approach become less significant, and it is now becoming practical to consider the application of the particle system approach to structural analysis. The following discussion describes a project to apply the particle system approach to structural engineering problems. The discussion takes the following steps:

- Describe and compare the computational approaches and user interfaces traditionally used in structural engineering and in computer graphics particle systems.
- Describe an implementation prototype, called *Arcade*, which applies the particle system approach to engineering problems, presenting illustrative examples and verifying engineering accuracy.
- Identify potential impacts of the particle system approach on education and practice in structural engineering and architecture.

2. STRUCTURAL ANALYSIS BACKGROUND

The objective of any structural analysis is to determine the response of a structure to an action, where the action may include applied forces, support movements, temperature changes, etc. One of the key characteristics of analysis is the treatment of time. Analysis can be categorised into *static* analysis, which completely neglects time, or *dynamic* analysis, which accounts for time effects. Dynamic analysis can further be broken down into *time history* analysis, which determines the response of a structure at a series of points through time, or *modal* analysis, which determines the response of a structure as a combination of harmonic modes of vibration.

Another key characteristic of analysis is whether it is *linear* or *non-linear*. Linear analysis assumes that the response of a structure is proportional to the action; e.g. if the load on a structure is doubled, then the stresses and movements of the structure also double. In general, linear analysis is highly unrealistic, because there are many important phenomena which result in non-linear behaviour, these include the following:

- **Large displacements**: Linear analysis assumes that differences between the geometry of the deformed and undeformed structure are negligible.
- **Material yielding**: Linear analysis assumes that material remains elastic under at all load levels, without permanent deformation or fracture.
- **Contact phenomena**: Linear analysis assumes that all connections work equally well in tension or compression, however this assumption does

not hold for connections where one object simply bears against another, since bearing can transfer compressive stress by not tension.

- **Buckling**: Linear analysis assumes that all members work equally well in tension or compression, neglecting the tendency of slender members to buckle in compression.

Structural engineering practice in building design extensively uses linear static and dynamic modal analysis methods. While these methods are effective in designing safe structures, they are less effective in simulating the real behaviour of structures; this is particularly true with respect to failure and collapse, which typically include any or all of large displacements, material yielding, contact phenomena, and buckling. Such simulation requires non-linear time-history analysis.

Although non-linear time-history analysis is still considered an advanced and somewhat exotic method in structural engineering practice, it is commonplace in the world of computer animation. For animation, time history analysis is essential to calculate the state of the subject at each animation frame. Non-linear analysis is necessary to model the large displacements that naturally occur with the subjects in an animation. In addition, interactive computer games require that the analysis be done in real time, so that the structure can respond immediately and realistically to input from the player.

Although the particle system approach has several shortcomings as a general-purpose method for structural analysis, its ability to support non-linear, real-time, interactive analysis opens new possibilities, particularly in providing physical insight into highly non-linear phenomena through a very different user interface model.

3. USER INTERFACE MODELS IN STRUCTURAL ANALYSIS SOFTWARE

From the early days of computer punch cards to the present day of graphic user interfaces, the conceptual model for structural analysis has retained the following three fundamental stages:

1. **Modelling**: Create a numeric model representing the structure and the actions on it.
2. **Analysis**: Perform the calculations to determine the structure's response to the actions.
3. **Review**: Examine the results of the calculation describing the response of the structure to the actions.

In the early days of structural analysis, the modelling phase consisted of manually preparing tables of numbers for input, and the review phase

consisted of scanning tables of numbers from the computer printout. Over the past fifteen years, the modelling and review stages have become far more graphic and interactive, and the analysis stage has become much faster and able to accommodate larger and more complex structural models. Despite those enormous improvements, the process has retained its three-stage batch-processing organisation: modelling, analysis, review, corresponding to input, process, output. In this organisation, seeing the effect of a change to the structure or its loading requires modifying the input, repeating the analysis, and repeating the review of the output. There is interactivity within the stages of modelling and review, but not in the modelling-analysis-review cycle.

It is useful to compare this model with that of interactive computer games. In games, the modelling-analysis-review cycle runs in a constantly repeating loop of real-time computation. Changes to the model are implemented through player actions with a mouse or other controller. The response of the structure is analysed immediately and its display on the screen corresponds to the review. The cycle runs at sufficient speed to provide a graphic update of the analysis results 15 to 30 times per second.

The following discussion describes an implementation prototype, called *Arcade*, whose objective is to merge the computational approaches commonly used in computer games and structural engineering, achieving the real-time interactivity of games while incorporating more sophisticated elements than simple springs, plus sufficient accuracy for engineering applications.

4. COMPUTATIONAL APPROACH

A particle system models a structure as a collection of particle masses connected by springs. The springs will be called *elements* in this discussion, consistent with terminology in structural engineering, and the particles will be called *nodes*. The analysis uses a time-step simulation, where the position and velocity of each node are known at the beginning of each step. The acceleration at the beginning of the step is calculated by first calculating the forces on the node, and then dividing by the node's mass. This calculation is done for each of the node's degrees of freedom, which for this two-dimensional application are horizontal and vertical translation plus rotation. Figure 1 shows the calculation loop performed with each time step:

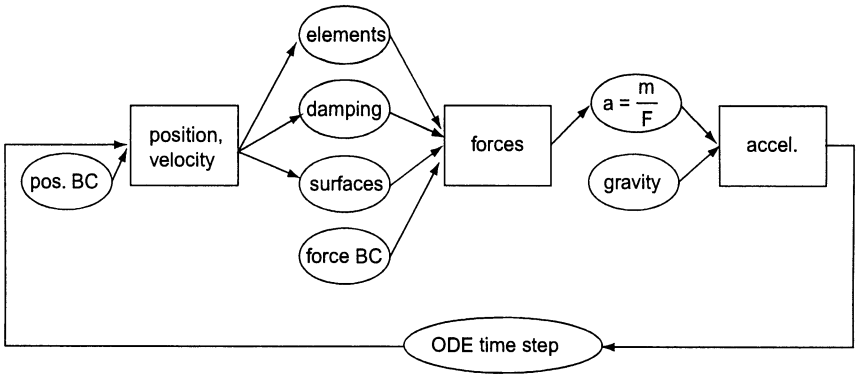


Figure 1. Calculation loop for a particle system The calculation begins with nodal position and velocity at the beginning of the step, then calculating forces, then calculating acceleration, then moving to position and velocity for the next step.

The figure shows that forces on a particle may arise from several different sources, the following are examples included in the *Arcade* implementation (several others are possible):

- **Elements**: Element forces arise from distortions of elements to which the particle is connected. Elements may be simple linear springs connecting two particles, where the force is proportional to the change in spring length, or more complex beam-type elements which transfer shear and moment, and model material yielding.
- **Damping**: Damping forces can be applied directly to a node based on its velocity. With high damping, the nodes behave as if they are moving through a thick fluid.
- **Surfaces**: It is relatively simple to model an elastic planar surface such as the ground to support particles using a *soft contact* approach (Cundall and Hart, 1992) where forces are applied to the particle based on its penetration into the surface. More sophisticated models can include friction and adhesion.
- **Force Boundary Condition**: Although rarely used in computer games, structural engineering commonly uses force boundary conditions where a force is applied to directly to a structure as an external load (e.g. applying a 100 kilo-Newtons force directly to a particle).

When all the forces are calculated on each particle, the acceleration of each particle can be calculated using Newton's second law, dividing the mass of the particle by the acceleration for each translation degree of freedom, and the mass moment of inertia for the rotational degree of freedom. When the elements are simple springs which work only in tension of compression, it is not necessary to consider rotational degrees of freedom, since particle rotation does not effect the spring force. For flexural beam-

type elements, which curve and bend, it is necessary to consider angular acceleration and mass moment of inertia.

With the acceleration of each degree of freedom known, the velocity and position at the next time step can be solved using numeric methods for solving the system ordinary differential equations. In concept, this can be solved using Euler's method as follows:

$$v_{n+1} = a_n \Delta_t$$
$$p_{n+1} = v_n \Delta_t$$

Where a_n, v_n, and p_n, are the acceleration, velocity, and position respectively of a nodal degree of freedom at time step n, and Δ_t is the time step increment. The calculation of the position for the next time step also accounts for displacement boundary conditions imposed on nodes. Typically such boundary conditions are used to model structural supports where the movement of the node is limited in one or more degrees of freedom.

In practice, Euler's method is ineffective because it requires an extremely small time step to achieve sufficient accuracy, and other more sophisticated methods are commonly applied (Witkin and Baraff, 1997a). The Arcade program uses an implicit method based on the trapezoid rule of integration, requiring the system of differential equations to be evaluated three times per time step. Simulation using the particle approach typically requires a time step increment in the range of 0.5 to 0.1 milliseconds. With three evaluations of the equations per time step, that increment range means the program will typically evaluate the system of equations 6000 to 30000 times per second.

5. EXAMPLES AND VERIFICATION

5.1 Interactivity

Figure 2 shows a multiple-exposure screen shot which illustrates the interactive nature of the program. The figure shows a truss structure which is supported at the top and bottom joints to the far left. The mouse is used to select the node at the lower right, the mouse is then clicked and dragged. The dragging adds a temporary element which links the selected node to the mouse. As the mouse is moved, the node at the link follows it, exerting a force on the structure.

Note the use of a linking element to exert forces on the node with the mouse rather than using the mouse directly. When the mouse is used directly on a structure where the elements are very stiff, the high velocities and

accelerations of the mouse can generate extremely large forces in the members and lead to numeric instability. Using a relatively flexible link element provides an effective cushion for the mouse forces.

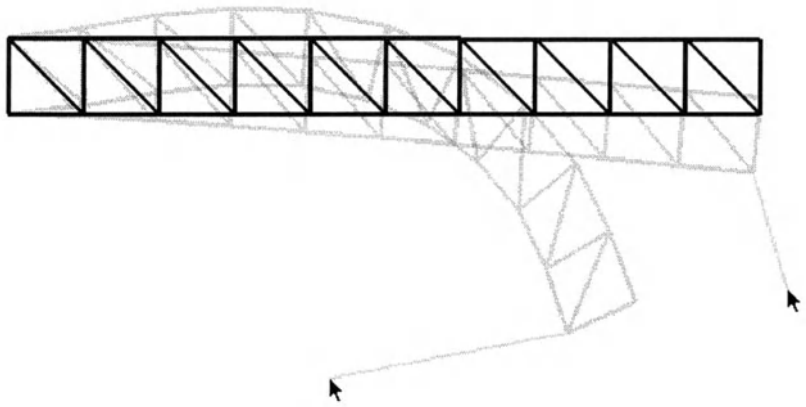


Figure 2. Truss structure loaded interactively via the mouse. The black outline shows the unloaded configuration. The grey shows steps of loading.

In addition to applying forces with the mouse, the program also allows the mass of a node and the global damping to be manipulated interactively. Many other options are possible. The computational approach allows any property of the model to be modified in real time through sliders or other devices and for the effects on structural behaviour to be displayed immediately.

5.2 Element types

One of the primary goals of the program is to introduce element types which are commonly used structural engineering, in particular elements that model material yielding. The elements implemented in Arcade are similar to those used in the DRAIN-2D program developed by Powell (1973), one of the earliest applications of non-linear analysis in structural engineering and still widely used for non-linear earthquake analysis of building frames. Figure 3 shows a the force-deformation characteristics of the non-linear truss element used in the example of figure 2.

Like a simple spring, the element resists only tension and compression forces along its axis, depending on its change in length. Unlike a simple spring, the element can model permanent deformations when the load exceeds a specified yield load (indicated by P_y on the graph). After yielding,

the bar unloads on the elastic stiffness, and then yields again on load reversal as shown on the graph.

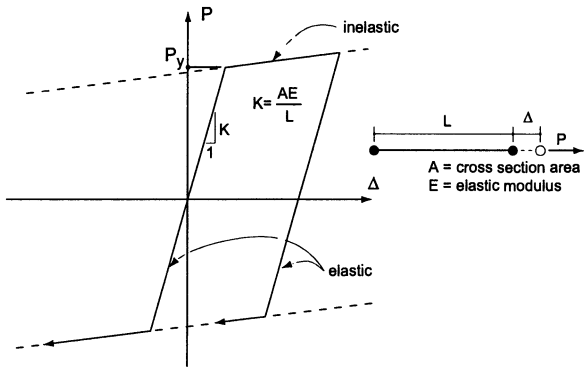


Figure 3. Force-deformation relationship for non-linear truss element

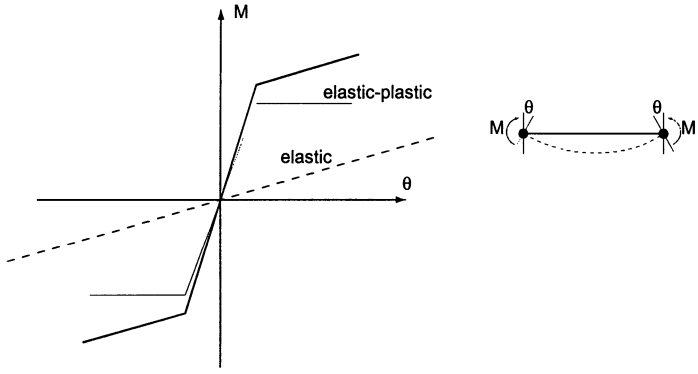


Figure 4. Force-displacement relationship for flexural component of beam element

The other element is a beam-column element which, in addition to resisting axial tension and compression, also resists shear and moment, which depend on the bending of the element (this component is called the *flexural* component). The flexural component also includes material yielding as shown in figure 4. The flexural behaviour is modelled by using two element components working in parallel: a completely elastic component, and an elastic-plastic component. Together, the two elements simulate an initial elastic range, followed by a plastic range with hardening. This concept is based on the DRAIN-2D beam element (Powell 1973), although it is simplified since it does not account for axial yielding, or for interaction between axial and flexural yielding. Figure 5 illustrates the behaviour of the inelastic beam element, as well as a contact surface. The figure shows

frames from an animation where a circular structure viewed in elevation falls under the force of gravity and strikes a horizontal surface. The impact with the surface effectively crushes the structure. Note the elements rendered in grey in the third and fourth frames, indicating that the element is on the inelastic branch of it's load-deformation curve.

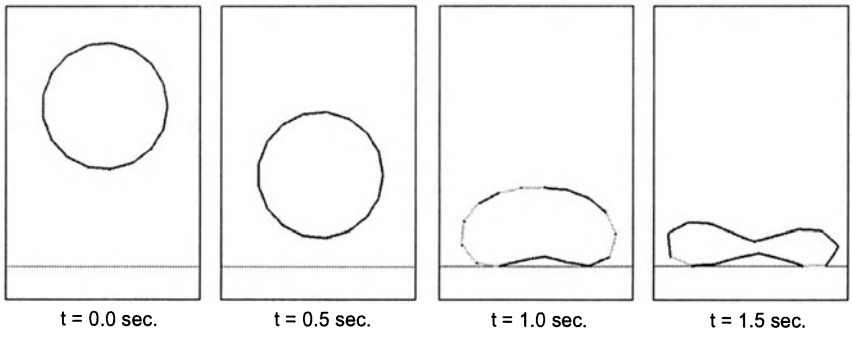


Figure 5. Circular structure, modelled with inelastic beam elements, falling under gravity and striking a horizontal surface. The grey rendering of some elements indicates that they are on the inelastic branch of the load-deformation curve.

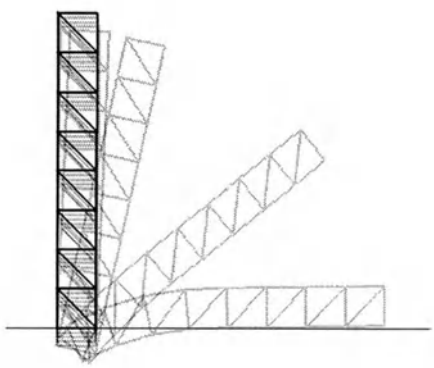


Figure 6. A tower sinking under its own weight into a viscous medium. The exposures are taken at 5-second intervals

The example from figure 5 illustrates the ability of a contact surface to simulate a stiff surface. In addition to stiffness, these surfaces can also model viscosity. Figure 6 shows a multi-exposure rendering of a tower structure bearing on a surface which has no elastic stiffness, but high viscosity, simulating the effect of the tower sinking into soft mud under its

own weight. When viscosity is used in combination with elastic stiffness, the surface becomes sticky, similar to a wall or floor coated with adhesive.

5.3 Engineering accuracy

Computer graphics and games naturally emphasise visual qualities over physical accuracy (Blum and Thumrugati 1997), and some applications deliberately produce inaccurate results by reducing the material stiffness far below realistic values (O'Brien and Hodgins 1999), e.g. modelling a metal as if were hard plastic. The stiffness reduction has two advantages: first, it reduces the velocity and acceleration of deformations, making the action easier to see; and second, it allows the simulation calculations to remain stable with a much larger time step, reducing the computational effort required, and allowing more complex models to be rendered at interactive frame rates.

One of the challenges of applying the particle system approach to engineering problems is achieving accurate results on realistically stiff structures at interactive frame rates. Figure 7 shows a verification example based on the Euler buckling formula. The figure shows a column which is fixed against all movement at the base, and restrained against horizontal movement at the top. The column is subjected to a gradually increasing vertical load, plus a moment at the top equal to 0.02 times the vertical load. The purpose of the small moment is to introduce slight curvature into the column, since a perfectly straight column can carry load beyond the buckling limit. It is modelled using eight elements of equal length, and nine nodes. The structural steel column has the following properties: length = 12.7 m (500 in.) area = 400 cm^2 (62 in^2), moment of inertia = 42870 cm^4 (1030 in^4). The Euler buckling load is then 10.7 MN (2406 kips). When the column is loaded over a period of ten seconds, it exhibits clear buckling deformations at 103% of the theoretical buckling load, and extremely large deformations at 104% of the buckling load. The deformed shapes are shown in the figure. The accuracy of the results are encouraging, although the stiffness of the structure required a relatively small time step of 0.1 milliseconds in order to maintain numeric stability.

Figure 8 shows another example using the inelastic beam element to compared results with simple plastic theory. The figure shows a beam which is completely fixed at the left end, and restrained against vertical movement at the right end, with a concentrated load applied at mid-span. The beam is modelled as elastic perfectly plastic, meaning that it has zero stiffness in the inelastic range. The beam the following properties: span 12.7 m (500 in.), moment of inertia 112661 cm^4 (2750 in^4), area 272 cm^2 (42.1 cm^2), plastic

modulus 5277 cm^3 (322 in^3), yield stress 345 MPa (50 ksi). The analysis model uses six equal-length elements and seven nodes.

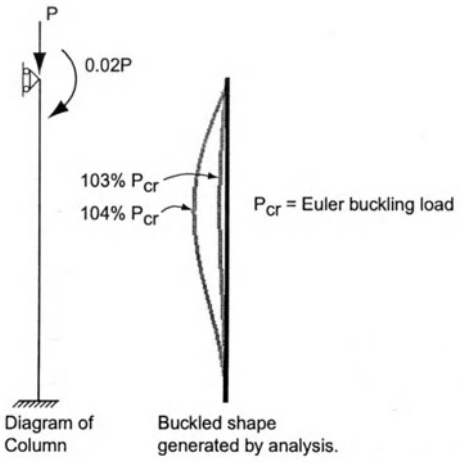


Figure 7. Euler buckling of a column, fixed at the base and pinned at the top. The deformed shapes show the position of the column at 103% and 104% of the Euler Buckling load

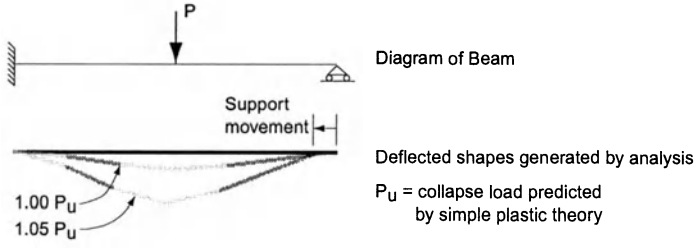


Figure 8. Beam structure with a fixed support at the left end and vertical movement restrained at the right end. The inelastic structure stabilises at 105% of the collapse load because the span shortens by 5%. (note that the displacements are not magnified)

According to simple plastic theory, the collapse load for this beam is 860 kN (193 kips) (Neal 1977). In analysis, when the beam is slowly loaded to the collapse load, the collapse mechanism forms as expected, initiated by yielding at the support, followed by yielding at mid-span. In theory, when both these regions have yielded, the beam becomes a mechanism and cannot resist further load, but the analysis shows that the beam does not collapse, it stabilises, continuing to carry the load. When the analysis increased the load to 5% greater than the collapse load, the beam experiences very large displacements, but does not collapse completely. This result seems at odds with the prediction of simple plastic theory, until the deformed shape of the

beam is examined closely. As figure 8 shows, the deformation of the beam causes the right end of the beam to move to the left, effectively shortening the span.. At 5% over the collapse load, the displacement of the right support is 68.6 cm (27 in), which is 5.4% of the total span. This shortening of the span increases the beam's ability to carry load proportionately. Taking the deformation into account, which simple plastic theory normally does not do, the results are quite sensible.

6. POTENTIAL IMPACT

The primary strength of interactive non-linear time-history analysis is that it is the analysis method which most closely mimics real life: reality is an interactive, non-linear, time-history. It's greatest potential is in giving a truer picture of the non-linear behaviour of structures. In the engineering profession, this type of analysis can give designers deeper understanding of a structural concept, although it will certainly not replace static analysis and modal dynamic analysis, which are efficient and effective design tools for many tasks. The particle system approach has the potential to make non-linear time-history analysis a more commonplace, mainstream method which can join more conventional design tools in the practising engineer's toolbox.

There are potentially more profound impacts in structural education in engineering and architecture. In engineering, non-linear, dynamic analysis is typically not taught at the undergraduate level, and only sparingly at the graduate level, and is typically not included in architectural education at all. Yet, many important structural phenomena, particularly those related to collapse, are non-linear and dynamic. Interactive non-linear analysis has the potential to enable structural education emphasise understanding structural behaviour, bring deeper meaning to the study of analysis procedures and building code requirements.

7. EFFICIENCY ISSUES

For many problems, non-linear time history analysis is monumentally inefficient. In any problem that can be solved by static linear theory, such as the deflection of a stiff elastic beam under load, the computer essentially needs to perform one solution of the equilibrium equations to determine the structural response. Using non-linear time history, the solution to such a problem could take three to four seconds of simulation time, with a time step in the range of 0.1 to 0.5 milliseconds, performing three solutions of the equations per time step. Determining the response, then, requires solving the

system of equations in the range of 18000 to 120000 times, compared to one solution for a linear static analysis.

This gross inefficiency is certainly one of the reasons that the particle system approach was not adopted in the early days of computer-based structural analysis, because computers were weak and processing cycles were scarce. The processing demands of the method made it completely impractical. The rapid growth of computer power now creates a radically different environment. Common computers are able to perform a complete static analysis in fraction of second for realistic structural models. As computing power continues to grow exponentially, computational efficiency will become a decreasingly important factor in choosing an analysis strategy.

8. FUTURE WORK

The key areas for future development are expanding to three dimensions and incorporating more sophisticated elements. For the truss element, the expansion to three dimensions is quite simple, since the element neglects rotations of the nodes. Rendering is also simple in three dimensions, since the system uses the OpenGL rendering library, which is already three dimensional. The key area for development is in the three dimensional beam element, which becomes much more complex since it must account for the three-dimensional rotation of the nodes.

Another area for development are the interactivity controls. Grabbing structures with the mouse is easy to understand, but offers very little control. Sliders, dials, and haptic force-feedback devices offer opportunities for more controlled interaction.

9. SUMMARY

Many computer games now incorporate real time simulation of physics in order to achieve visual realism. As computational speed and capacity continue to increase, it is reasonable to expect that the physical simulations in computer games will continue to become more sophisticated, and not far in the future, computer games will probably incorporate advanced engineering finite element methods simply for visual realism (e.g. O'Brien and Hodgins 1999). As the gaming industry borrows methods from engineers, it is reasonable for engineers to consider what they can borrow from the gaming industry. One of the lessons to learn from games is the importance of interactivity, and the ability to perform real-time, non-linear

physical simulation. Such analysis will clearly not replace other methods used in structural engineering, but has the potential to complement them, adding an important tool to the structural engineering repertoire. The potential impact is particularly important in structural education, which currently places little emphasis on the non-linear and dynamic phenomena that commonly characterise structural failure.

10. REFERENCES

Blum, Mike, U. Thumrugati, 1997, "Using Dynamics in Disney's Production Environment", presentation slides for Siggraph '97 Course *Physically Based Modeling: Principles and Practice. http://www.cs.cmu.edu/afs/cs/user/baraff/www/sigcourse/slidesg.pdf*

Cundall, Peter A., R.D. Hart, 1992, "Numerical Modelling of Discontinua", *Engineering Computations*, vol. 9, p. 101-113.

Hecker, Chris, "Physics, The Next Frontier", 1996, *Game Developer,* October/November, p. 12-20.

Neal, B. G., 1977, *The Plastic Methods of Structural Analysis*, Chapman and Hall, London.

O'Brien, James F., J. K. Hodgins, 1999, "Graphical Modeling and Animation of Brittle Fracture", *Proceedings of the ACM Siggraph Conference on Computer Graphics*, p. 137-146.

Powell, Graham H., 1973, *DRAIN-2D User's Guide*, UCB/EERC-73/22: Earthquake Engineering Research Center, University of California, Berkeley.

Witkin, Andrew, D. Baraff, 1997a, "Differential Equation Basics" lecture notes for Siggraph 1997 Course *Physically Based Modeling: Principles and Practice http://www.cs.cmu.edu/afs/cs/user/baraff/www/sigcourse/notesb.pdf*

Witkin, Andrew, D. Baraff, 1997b, "Particle System Dynamics" lecture notes for Siggraph 1997 Course *Physically Based Modeling: Principles and Practice http://www.cs.cmu.edu/afs/cs/user/baraff/www/sigcourse/notesc.pdf*

Simulation and representation

Learning from airflow analyses in buildings

Alexander Koutamanis and Peter den Hartog
Faculty of Architecture, Delft University of Technology

Key words: simulation, representation, visualization

Abstract: The simulation of environmental aspects is a current priority in design research and practice. The availability of relatively efficient and reliable simulation systems and the emphasis on environmental aspects throughout a building's lifecycle combine to stimulate exploration of aspects such as lighting and air quality by computational means. Nevertheless, a frequent complaint is that the addition of such simulations makes design information processing time-consuming and cumbersome, thereby increasing uncertainty and indecision. Therefore, it is imperative that simulation is integrated in the strategies and tools normally used by the digitally-minded architect. In this respect a central issue is the relations between the simulation and the design representation used as connecting tissue for the whole design environment. Input of design information in the simulation means identification of relevant objects, aspects, parts and properties of these objects, as well as relationships between objects. The explicit description of objects such as spaces, doors and windows in the design representation allows for ready extraction of relevant information, including automatic recognition of relationships such as adjacency between a window and a space. The addition of information specific to the airflow analysis was resolved by the extension of the representation to cover front-end service components such as inlets and outlets and general properties (annotations) such as activities accommodated in a space and the primary choice of cooling and heating subsystems. The design representation is also the obvious target for the output of the simulation (feedback). Visualization of airflow in terms of the resulting voxels makes effortless and enjoyable viewing but merely allowing the visualization to coexist with the representation of spaces and building elements does not provide design guidance. One way of achieving that is by treating spaces not as integral entities but as containers of relevant surfaces. These surfaces determine the adaptive subdivision of the space and function as attractors for voxel clustering.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 657–666.
© 2001 *Kluwer Academic Publishers.*

1. SIMULATION IN CONTEXT

The simulation of environmental aspects is a current priority in design research and practice. The increasing consciousness of indoor climate problems and possibilities, as well as the multiplicity of human activities and their intensive technological support, is leading to a growing number of programmatic and legal requirements with a rising specificity. The indoor climate of a building is the product of active conditions such as sun and occupant activity and passive building features such a window area and shape, natural shading and material properties. Unfortunately indoor climate is at best reduced to simplistic rules of thumb while the production of healthy buildings presupposes monitoring and control of several qualities of indoor climate throughout the design process. In optimal cases the passive building features are configured in a manner that results in comfortable indoor climate with temperature, air velocity and air purity within prescribed ranges. More often than not however, they exceed ranges that are considered healthy and additional cooling, heating or ventilation is needed.

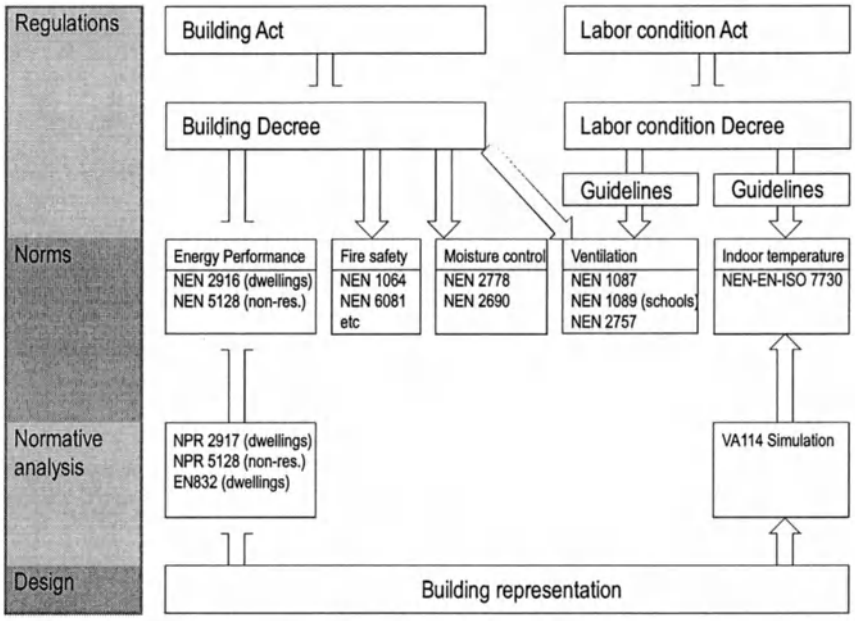


Figure 1. Requirements complexity: Dutch norms and regulations on indoor climate

The addition of building services further complicates the design of indoor climate. Employing standard solutions for these subsystems is no guarantee for acceptable results. A complete analysis and evaluation of the design situation is imperative for the design of both building features and indoor

climate. Current design practices separate analysis and synthesis in terms of design stage and participation of various parties. Design synthesis is the domain of the architect, while consulting engineers carry out analyses and performance calculations the results of which are fed back to the designers.

The distance between architect and analysis may alienate designers from the possibilities and implications of the analysis. It may also degrade the cooperation between designers and consultants to unproductive confrontation. As a result, the relevance and accuracy of both the analysis and the feedback of results suffer. For example, architects regularly use general rules of thumb that may be incompatible with the precise mathematical models advising specialists employ. This means that specialists are frequently compelled to extract analysis input from their own interpretation of design documentation. The position is reversed later in the process, when architects face unfamiliar analysis data and formats.

One solution to both the communication problem between designers and specialist advisers and the lack of analysis in the early design stages relies on computer simulation. With the democratization of the computer, experts started recognizing its potential for solving efficiently and accurately the highly complex problems of indoor climate analysis. The methods and techniques that were developed in response to that managed to provide a stable basis for confronting realistic problems, as well as for further development. Paramount among these for indoor climate analysis is computational fluid dynamics (CFD). CFD is a technique to calculate patterns of fluid flow using a fundamental set of partial differential equations. These equations derive from three basic principles: conservation of mass, conservation of momentum and conservation of energy within that fluid. The equations are partial integral, non-linear and too complex to be solved analytically. For this reason, discretization and iteration are used to arrive at a solution that describes the characteristics of the moving fluid with specific numbers for velocity, temperature, pressures and other characteristics (Anderson, 1995; Wendt, Anderson et al., 1996).

The transition of CFD simulations from specialist to design tool is being accelerated by the increasing availability of computational power and by the rising interest in building and design performance. The integration of analysis by simulation into designing early on in the design process empowers the designer with predictions of building behaviour and performance that may be less accurate and precise than the ones a specialist could achieve but nevertheless provide valuable feedback and hence possibilities for design guidance (Mahdavi, Mathew et al., 1997; Mahdavi, and Suter, 1997; Papamichael, Laporta et al., 1997; Hartog, Koutamanis et al., 1998; Hartog, Koutamanis et al., 2000). In particular, the possibility of representing dynamically and visually processes of a dynamic nature such as

airflow that are experienced only fragmentary and tentatively is a major factor of increasing awareness of causes and effects in the indoor climate of a design. Even in the hands of novice designers it is instrumental in transforming trial-and-error improvisations to generate-and-test explorations of a design's variables (Hartog and Koutamanis, 2000). Moreover, the combination of qualitative and quantitative aspects in the presentation of analysis results lays the foundation of meaningful and productive communication between designer and specialist towards a mutual comprehension of the full extent of design problems and decisions.

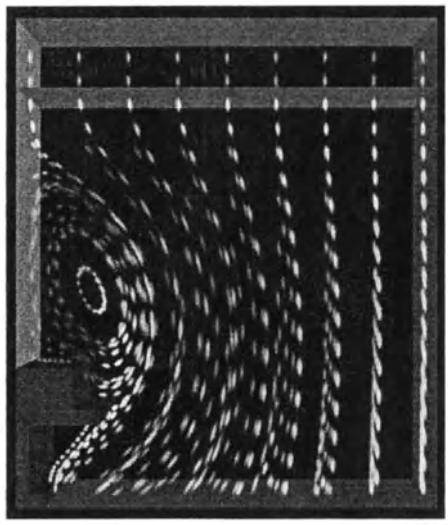


Figure 2. CFD simulation of airflow

2. REPRESENTATION AND INPUT

The integration of CFD simulation in the strategies and tools of the digitally-minded architects depends on a number of pragmatic factors. These include:

1. The availability of affordable simulation systems
2. The efficiency and reliability of simulation systems
3. The educational potential of simulation systems, including interface design and background information
4. The ease of inputting building information in simulation systems

The first two factors are a matter of market dynamics and can be expected to have a positive development with the growing demand for CFD simulation. The third factor relates more to architectural education and will probably remain one of the main shortcomings of commercial software

development. The fourth one is a purely methodical issue. Digital design information management is arguably one of the emerging hot items in architectural computerization and one that cannot be resolved by opportunistic solutions such as bilateral data exchange between programs or procrustean or gargantuan standardization schemes. Our working hypothesis is that the connecting tissue between different modalities, activities and aspects is a modular, hierarchical representation of buildings that permits transition from one aspect or abstraction level to another without loss of overview or design focus (Koutamanis, 1993; 1997; 2000).

Input of design information in the CFD simulation means identification of relevant objects, aspects, parts and properties of these objects, as well as relationships between objects. Such information is normally not explicit in a conventional CAD drawing or other digital design document. This is not a limitation of CAD programs but of the mindless reproduction of analogue design and drawing practices in the computer. Admittedly CAD programs offer little support for the flexible and efficient manipulation of meaningful design entities but this is something that can be easily alleviated by customization, i.e. the addition of purpose-built modules that structure and interpret the low-level geometric primitives of CAD (Koutamanis and Mitossi, 1993; 2000a). These facilitate the development of structured representations that cover a wide scope of design aspects and activities with minimal deviation from current professional practices (Mitossi and Koutamanis, 1998; Koutamanis and Mitossi, 2000b).

The explicit description of entities such as spaces, doors and windows in the design representation allows for ready extraction of relevant information. This refers to entity properties that can be derived from its geometry: location, shape, area and volume. Property recognition can be augmented by means of alphanumeric and graphic indications of non-geometric properties, such as color and material, which can be correlated with e.g. sizes derived from the geometry of the entity. Such information is essential input to simulations and analyses of indoor climate (Hartog, Koutamanis et al., 1998).

The advantages of a structured representation for the input to a CFD simulation go beyond what can be identified and measured in individual entities. Also relationships between entities can be recognized automatically. For example, the presence and position of openings in a space can be recognized by relationships of adjacency between a space and windows, doors or other openings. As a result, it is possible to automate input of information on the context of entities and global characteristics of a building, such as orientation, compactness and zoning.

Recognition of properties and relationships is facilitated by a modular organization of the entities in clusters defined by spatial properties such as

location in the building and intrinsic entity properties such as construction or functional type. This modularization of information can be achieved by means of layers. In our system these indicate the floor level and the functional type of the entities, such as "windows on the ground floor" or *00_window* in our coding convention.

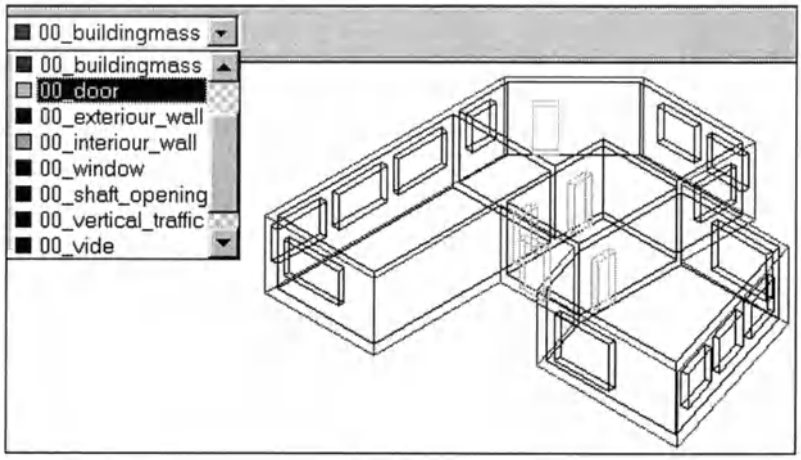


Figure 3. Modularization of design information by means of layers indicating floor level and entity type

The addition of information specific to the airflow analysis follows the same principles. The building representation is extended to cover front-end service components such as inlets and outlets and general properties such as activities accommodated in a space and the primary choice of cooling and heating subsystems. These are generally represented by alphanumeric annotations on spaces and building elements. When these components carry instance-specific information (properties) it is also possible to represent them with separate entities contained mostly in spaces.

In general lines, the use of the representation for inputting design data to the CFD airflow simulation followed the same principles and required similar structuring as other formal and functional analyses (Mitossi and Koutamanis, 1998). In particular, it echoed earlier experiences with light simulation. There too input to the simulation referred to integral, recognizable design entities, their properties and interrelationships. The only difference lies in the input range of each individual act of simulation: airflow simulation requires information on the wider context of a space that is not relevant to light simulation.

3. REPRESENTATION AND OUTPUT

The design representation was also the obvious target for the output of the airflow simulation (feedback). In normative situations feedback can take the form of alphanumeric annotations that describe relevant performance properties of a space. This, however, reduces the detail and specificity of the simulation to global, static measures that do not do justice to the potential of CFD. More specifically, such measures fail to depict the emerging climatic zoning in a space, as well as the dynamic nature of climatic phenomena.

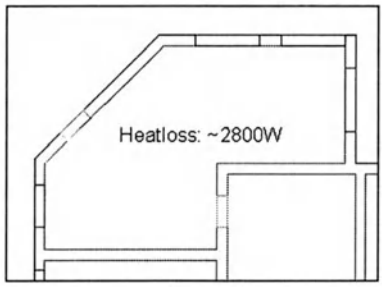


Figure 4. Normative feedback by means of annotations

The superimposition of analysis results, as in light simulation, is the obvious starting point for meaningful feedback to the design representation. In contrast to light simulation, however, the analysis results did not merely form graphic annotations of existing elements in the design representation. The finite elements quantization and calculation in the CFD subdivides a space by a uniform or adaptive grid that does not necessarily express fully its spatial articulation. This is generally viewed as an input and precision problem. Given the rapid progress of computational technologies, we may consider it as a mere temporary limitation that will be lifted by algorithmic improvement and added brute computational force.

In the accommodation of output, differences in the geometry of a space and that of the analysis results can be approached in two ways. The first relies on positive feedback that determines the abstraction of the space geometry by the structure and hence the required fineness of simulation results. This makes the analysis a cyclical process, with added advantages for the overall accuracy and relevance of the simulation. The second way relies on adaptive hierarchical subdivision techniques that are capable of organizing the simulation output using the form of the space as reference framework (Samet, 1990). The added advantage of the second way is the explicitness of airflow grain in different regions and, through that, recognition of climatic zoning in a space.

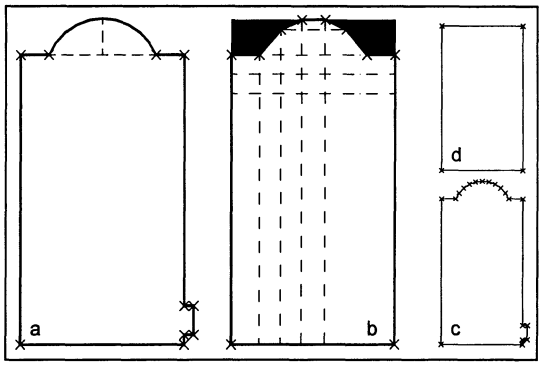


Figure 5. Form, geometry and abstraction: (a) input; (b) for CFD; (c) maximal precision; (d) global description

Feedback of dynamic aspects requires a finer subdivision of the space into voxels that can be used by techniques such as particles so as to give an impression of movement in the depicted qualities and quantities. The addition of such elements in the representation is demanding in computer power but otherwise straightforward and makes visualization of airflow effortless and enjoyable viewing. However, merely allowing the simulation results to coexist with the representation of spaces and building elements does not provide clear design guidance because it relies too heavily on human interpretation. Consequently, one can expect interference from spurious perceptual recognition and logical jumps, especially concerning causal relationships between analysis input and output.

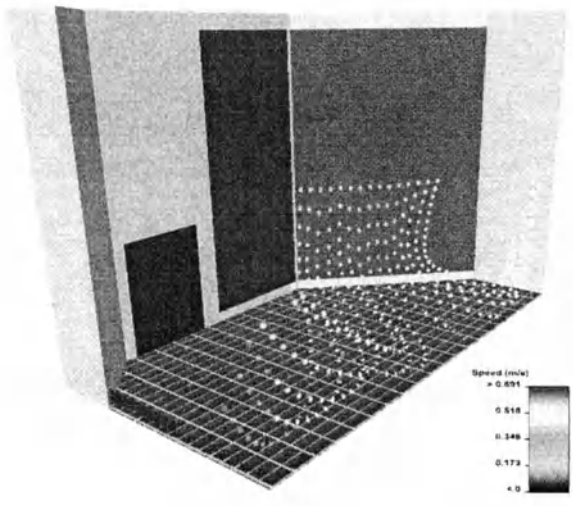


Figure 6. Particle visualization of airflow

4. REPRESENTATION REVISITED

The integration of simulation results into the design representation of a building suggests a need for elaboration of the representation beyond the level of apparent entities such as spaces and building elements, as well as their aggregates into conventional clusters such as wings or abstract coordinating devices (Koutamanis, 1996). One way of achieving that is by treating spaces not as integral entities but as containers of other, intermediate elements. From cognitive science we have evidence that surfaces form such intermediate elements between low and high level vision (Nakayama, He et al., 1995). These surfaces do not merely describe the internal structure of three-dimensional objects in a scene: surface recognition is relatively free from top-down, object-level knowledge and determines perceptual grouping and perceptual recognition. Hence it forms a prerequisite to rather than an effect of object recognition.

In the analysis of built space surfaces also play an important role. Our activities in the built environment relate strongly to dominant surfaces such as the ground, visual boundaries, the horizon and illusory contours. Such surfaces are contained in a space but in many cases they transcend individual spaces and may even unify or otherwise connect spaces with each other. In the specific case of CFD feedback to the design representation we can distinguish between:

- *Formal surfaces:* surfaces relating to the form of the building
- *Activity surfaces:* these accommodate human activities in the built environment and may coincide with formal surfaces, e.g. the ground.
- *Emerging surfaces:* aggregations of analysis results or relations
- *Subdivision boundaries:* products of subdivision schemes such as quadtrees in relation to analysis results

Such surfaces can be approached as reference structures that determine the organization and subdivision of the visual scene comprising the design representation and the visualization of the analysis. Activity surfaces are a pragmatic choice, as they relate climatic conditions to human interaction with objects such as work surfaces and desktops. Formal surfaces have the added property of constraining and modifying the behavior of airflow. Emerging surfaces and subdivision boundaries can be tentatively related to climatic changes in a space. In short, these surfaces arguably define the interaction between the simulation of indoor climate and the representation of its spatial form. Therefore, they are a key element in the formulation and interpretation of the analysis, e.g. as attractors for voxel clustering, and as such deserve explicitness in the representation.

5. REFERENCES

Anderson, J.D. (1995). *Computational fluid dynamics; the basics with applications.* New York, McGraw-Hill.

Hartog, J.P.d. and A. Koutamanis (2000). Teaching design simulation. in D. Donath (ed.) *Promise and reality.* Weimar, eCAADe & Bauhaus-Universität Weimar.

Hartog, J.P.d., A. Koutamanis, et al. (1998). Simulation and evaluation of environmental aspects throughout the design process. in *4th Design and Decision Support Systems in Architecture and Urban Planning Conference.* Eindhoven.

Hartog, J.P.d., A. Koutamanis, et al. (2000). Possibilities and limitations of CFD simulation for indoor climate analysis. in *Design and decision support systems in architecture. Proceedings of the 5th International Conference.* Eindhoven, Eindhoven University of Technology.

Koutamanis, A. (1993). "The future of visual design representations in architecture." *Automation in Construction* 2(1): 47–56.

Koutamanis, A. (1996). Elements and coordinating devices in architecture: An initial formulation. in *3rd Design and Decision Support Systems in Architecture and Urban Planning Conference. Part One: Architecture Proceedings.* Eindhoven.

Koutamanis, A. (1997). Multilevel representation of architectural designs. in R. Coyne, M. Ramscar, J. Lee and K. Zreik (eds.) *Design and the net.* Paris, Europia Productions.

Koutamanis, A. (2000). "Digital architectural visualization." *Automation in Construction* 9(4): 347-360.

Koutamanis, A. and V. Mitossi (1993). Adding visual recognition to the capabilities of computer-aided design. in J.J. Connor, S. Hernandez, T.K.S. Murthy and H. Power (eds.) *Visualization and intelligent design in engineering and architecture.* London / Southampton, Elsevier / Computational Mechanics Publications.

Koutamanis, A. and V. Mitossi (2000a). Grammatical and syntactic properties of CAAD representations for the early design stages. in *Design and decision support systems in architecture. Proceedings of the 5th International Conference.* Eindhoven, Eindhoven University of Technology.

Koutamanis, A. and V. Mitossi (2000b). "On representation." *Design Systems Reports* 2000(1): 74-82.

Mahdavi, A., P. Mathew, et al. (1997). "Bi-directional computational design support in the SEMPER environment." *Automation in Construction* 6: 353-373.

Mahdavi, A. and G. Suter (1997). "On implementing a computational facade design support tool." *Environment and Planning B: Planning and Design* 24: 493-503.

Mitossi, V. and A. Koutamanis (1998). Spatial representations as the basis of formal and functional analysis. in *4th Design and Decision Support Systems in Architecture and Urban Planning Conference.* Eindhoven.

Nakayama, K., Z.J. He, et al. (1995). Visual surface representation: a critical link between lower-level and higher-level vision. in S. M. Kosslyn and D. N. Osherson (eds.) *Visual congition. An invitation to cognitive science. 2nd ed.* Cambridge, Massachusetts.

Papamichael, K., J. LaPorta, et al. (1997). "Building Design Advisor: automated integration of multiple simulation tools." *Automation in Construction* 6(4 August): 341-352.

Samet, H. (1990). *The design and analysis of spatial data structures.* Reading, Massachusetts, Addison-Wesley.

Wendt, J.F., J.D. Anderson, et al. (1996). *Computational fluid dynamics: an introduction.* Berlin ; New York, Springer.

Performance-based computational design via differential modeling and two-staged mapping

Ardeshir Mahdavi, Rohini Brahme, Smita Gupta
School of Architecture, Carnegie Mellon University

Key words: building performance simulation, homology-based mapping, intelligent design agents

Abstract: Computational performance-based building design support faces a conflict. It is important to provide building performance feed back to the designer as early as possible in the design process. But many aspects of building performance are significantly affected by the design of the building's technical systems, which are typically configured in detail only in the later stages of design. The challenge is thus to find a method to use detailed simulation tools even during the early stages of design when values for many of the variables for the building's technical sub-systems are not yet available. In this paper, we demonstrate how this problem can be partially solved by combining two levels of automation. The first level consists of differential building representation involving a number of domain (application-specific) object models that are derived from a shared object model automatically. The second level uses generative agents that create reference designs for the technical sub-systems of the building. To demonstrate the feasibility of the proposed approach, we use the building energy systems domain (heating, cooling, ventilation, and air-conditioning) as a case in point.

1. INTRODUCTION

Certain levels of building performance analysis may be more relevant to the types of questions that primary building designers (particularly architects) must explore. Such questions address, for example, the effects of building enclosure and glazing, massing, orientation, natural ventilation. Performance analysis of detailed building technical sub-systems is usually associated with the activities of the domain specialists (e.g., lighting, energy, and acoustic experts). However, buildings' overall performance is

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 667–680.
© 2001 *Kluwer Academic Publishers.*

considerably affected by the design of the technical sub-systems. This circumstance poses a challenge to the developers of building performance analysis tools: How can we provide performance feed-back to the designer as early as possible in the design process, while considering the effects of technical sub-systems as well? The challenge is thus to find a method to use detailed simulation tools even during the early stages of design when information on building's sub-systems is either not available or is schematic at best. Such a method could deepen and extend the type and range of building performance queries that a general user (for instance, the primary building designer) could pursue early on in the design process.

Our approach toward a partial solution to this problem involves differential modelling and two levels of mapping:

1. Differential building representation - A general building representation (SOM, or shared object model) incorporates most of the information needed for the configurational definition of the building early in the design process (Mahdavi 1999). The detailed information on building's technical systems is captured in various disciplinary representations (DOMs, or domain object models). The first automated mapping operation (homology- based mapping) allows for the derivation of a basic DOM from SOM (Mahdavi and Wong 1998, Mahdavi et al. 1997). However, to perform computational building performance analysis, this basic DOM must be augmented with further detailed technical information. Such information is typically provided by the user. To do this without user intervention, a second automated mapping operation is needed.
2. Generative sub-system design agents - To perform the second automated mapping operation (from the initial DOM to a complete DOM), we adapt a technical sub-system design agent approach. As a case in point, we consider design agents for building's energy systems. We describe one such agent that automatically generates a layout for the building's energy delivery network (e.g., duct/pipe system) with minimal inputs from the user (high-level definition of the energy system type). Generally, the network design is undertaken during the later stages of design when most of the relevant decisions (e.g., location of the mechanical room, number of thermal systems, plenum size) have already been finalized. We demonstrate how a design agent can generate, in this case, a network lay-out using a combination of heuristics and shortest-path algorithm.

2. DIFFERENTIAL BUILDING REPRESENTATION

Building performance analysis can be performed at various levels of depth and resolution. From a software engineering point of view, this represents a number of problems. A building model that is too restricted, may allow only for a limited and ultimately less useful set of analysis options. On the other hand, a model that would capture all the requirements of technical sub-system analysis may become too large, leading to the classic problems of massive product models (Mahdavi et al. 1999). Our experience in this area has led us to a differential building representation approach. This approach distinguishes between a general building model and various building models for the different technical disciplines (Mahdavi 1999).

The general building model (SOM - shared object model) incorporates most of the information needed for the configurational definition of the building early in the design process (cp. Figure 1). It is the result of a bottom-up approach that began with the study of the informational requirements of a discrete set of building performance analysis applications. Two important criteria informed its development: *a)* the SOM should reflect a building representation that is transparent to building designers, and *b)* the domain object models for each of the applications should be derivable from the SOM without user intervention (Mahdavi et al. 1997).

The core technology to support this SOM-to-DOM derivation is a kind of homology-based mapping mechanism. Using it, it is possible, in principle, to make evolving building designs subject to comprehensive parametric studies in multiple domains without having to input the building model separately for each application (Mahdavi et al. 1998). Thus, building performance feedback may be provided to the user in an effective and timely fashion.

The detailed information on the building's technical sub-systems is captured in various disciplinary representations (DOMs - domain object models). Within the framework of the SEMPER project (Mahdavi 1999) such domain objects models have been developed for applications in energy and airflow analysis, lighting, acoustics, and environmental impact analysis.

Note that, the primary DOM that is derived from the SOM does not have, in most cases, the entire set of data needed for analysis. While basic passive energy analysis and daylight simulation may be performed based on respective primary DOMs, applications that involve extensive technical equipment and hardware (e.g. HVAC, electrical lighting) require additional information.

Figure 2 shows, as an example, the DOM of the HVAC sub-system. The highlighted boxes show the information that is automatically mapped from SOM. The HVAC simulation module utilizes a representation consisting of spatial units (cells) with nodes that define finite control volumes (Figure 3).

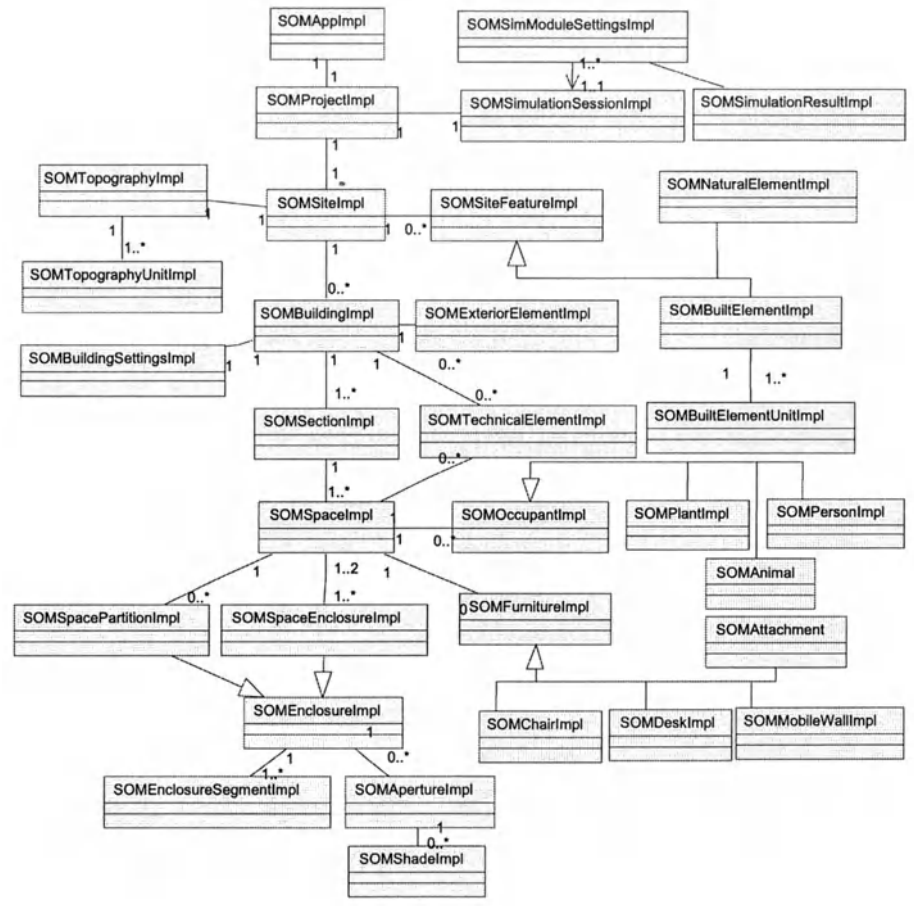


Figure 1. Shared Object Model (primary physical elements, UML notation)

A 3-dimensional grid for discretizing the spaces into cells is adopted, which serves as a framework for designing and modelling the HVAC distribution network and terminal unit design. The nodal representation of the building is configurationally homologous to the space-based building representation in the SOM and is automatically derivable from it. However, for a full HVAC analysis, two further conditions must be met:

i) The SOM must also incorporate some basic information for the various technical sub-systems to facilitate the second automated operation. In case of the HVAC sub-system, this consists of minimal information - the system type and location (to decide whether it is a pipe network or a duct network and its starting point) and the location of the distribution network (to decide if it is a ceiling based or a floor based system). Such information is readily available even during the early stages of design.

ii) Generative design agents must be developed that can carry out the second automated operation. They must be able to generate autonomously reference (default) designs of technical systems such as the layout of the energy distribution network. We discuss this in the next section.

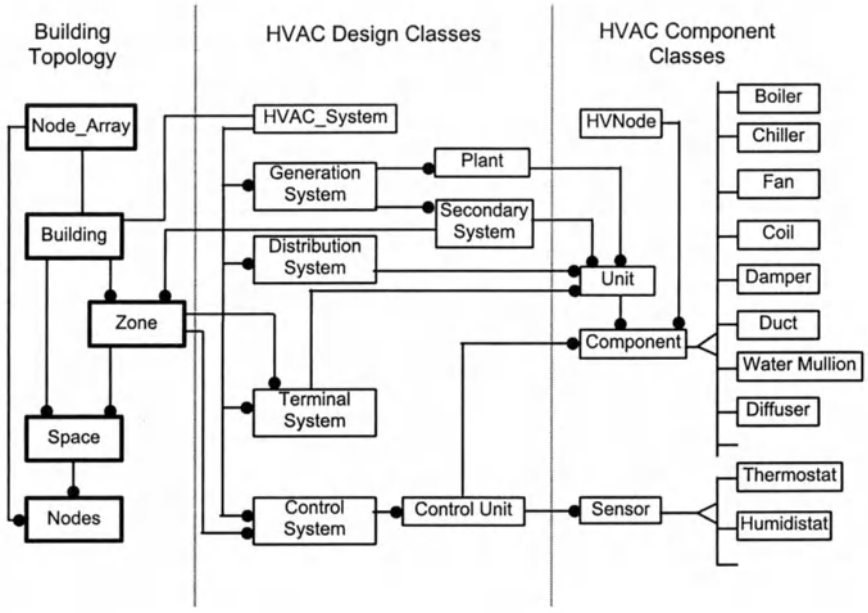


Figure 2. Domain Object Model

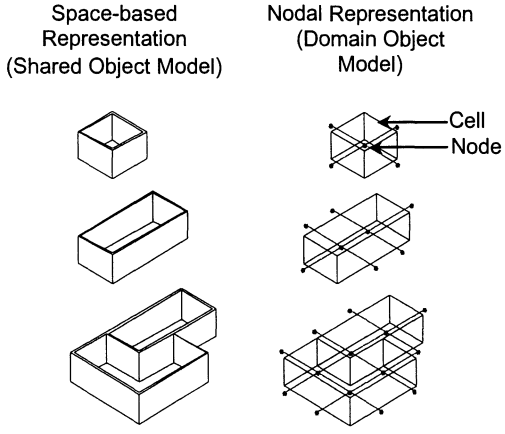


Figure 3. Homology-based mapping

3. TECHNICAL SUB-SYSTEM DESIGN

3.1 Overview

The first homology-based automated mapping operation allows for the derivation of a basic DOM from SOM. However, to perform computational building performance analysis, a complete DOM is needed. In case of the HVAC domain, such a complete DOM must entail a full description of the HVAC design classes and the component classes in order to allow for a performance analysis. To achieve this without user intervention, a second automated operation is needed - from the initial DOM to a complete DOM. Toward this end, we adapt a technical sub-system design agent approach.

Let us consider the case of HVAC domain again. Once the building design classes are obtained by the first mapping process (Figure 4, b), the remaining information needed to complete the HVAC DOM is generated automatically by using a generative design agent. Briefly, the design agent uses: *i)* the system type information to generate the components for the generation system; *ii)* certain heuristics to design the terminal system; *iii)* a combination of heuristics and shortest-path algorithm to design the distribution system components (Figure 4, c and d).

Once the system design is complete, all the components are sized automatically. These components are represented as nodes and their adjacencies with other components (or with boundary conditions) are represented as paths. The nodes and the paths make up the complete HVAC system network, which is numerically described by a system of equations formed by applying appropriate flow and energy equations to each node/path. Finally, the performance of the system (e.g., energy consumption) is calculated automatically at each time step, for the specified time frame.

3.2 Distribution System Design

Typically, the energy distribution system of a building conveys a heating or cooling medium from the generator location to the portion of a building that requires conditioning. It is unlikely that the primary building designer (typically the architect) would model different system configurations (central vs. distributed, floor-based vs. ceiling-based, ducted vs. plenum, etc.) to see their affect on energy consumption, if the network has to be designed and manually entered into the tool for each configuration. Thus, it seems that an automatically designed network could be useful in evaluating such alternatives, especially at the initial design phase.

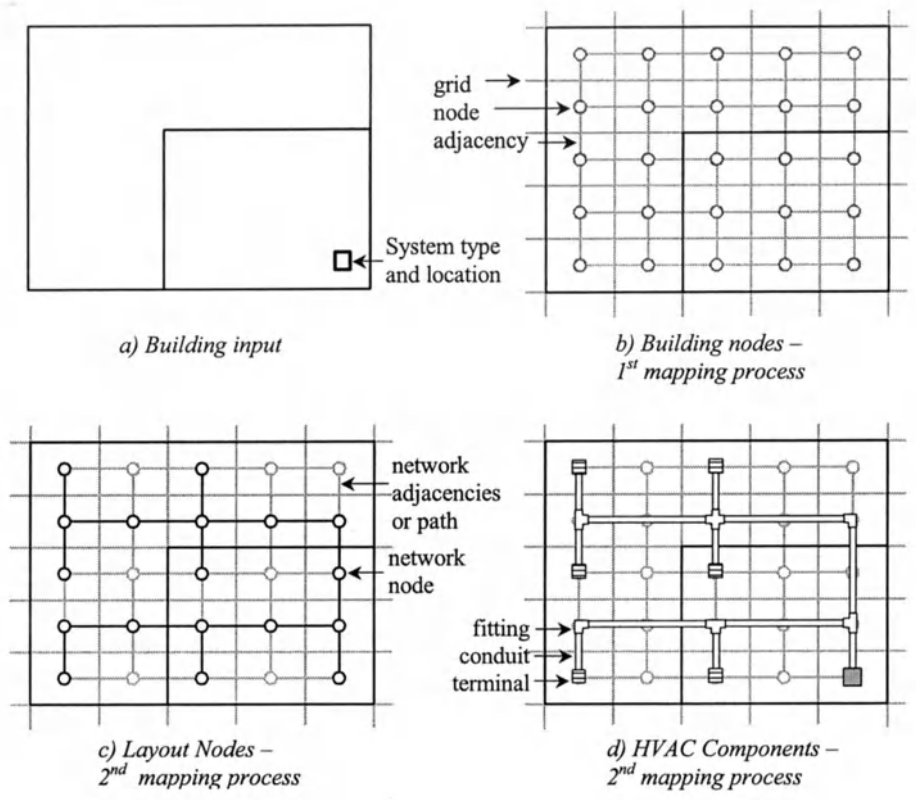


Figure 4. General scheme for grids, nodes, and component network structure

In our implementation, the design agent used for the automated generation of reference (standard or default) distribution networks uses a combination of heuristics and shortest-path algorithm. The inputs for the design agent are the building geometry, location and type of HVAC system and the location of the distribution network (whether it is floor-based or ceiling-based). This information is captured in the SOM and automatically mapped to the DOM in the first mapping operation. We can now proceed with the description of the second automated process and the distribution system design agent's functionality. This we do by establishing a terminology, summarizing the underlying heuristic rules, and describing the relevant algorithm.

Terminology: For a better understanding of the functionality of distribution system design agent, the following terms should be kept in mind.

- *Zone:* A group of spaces in a building which are controlled by the same controller.
- *Cluster:* A region in a zone encompassing all the terminals located within a certain range.

- *Branch:* A branch is a duct or pipe section that allows fluid flow from one point to another.
- *Start node:* The building node at the interface of the vertical distribution shaft with the floor.
- *Secondary system:* The secondary system equipment consists of prime movers (fan, pump), and heat and mass transfer components (coils, humidifiers, dehumidifiers). Secondary systems serve one or more sections of the building.
- *System node:* The building node at the location of the secondary system.
- *Plenum:* The set of building nodes at each floor (or level) through which ducts or pipes can pass.

Heuristic Rules: These are based on the analysis of the common practice of network design and the rules that are used by HVAC designers. It is assumed that all the spaces in a zone are contiguous.

1. The network should be as symmetric as possible.
2. There shall be a maximum of five levels of branches:
   - Terminal branch: Connects the terminals to the cluster branch
   - Cluster Branch: Centrally placed branch in the cluster to which terminals are connected.
   - Zone branch: Connects all the cluster branches in a zone.
   - Main branch: Connects all the zone branches to the start node on each floor.
   - Vertical branch: Connects the main branches on each floor to the system node
3. Branches in one zone cannot intersect with branches of another zone.
4. Main branch cannot intersect any zone branch.

Algorithm: The HVAC DOM's nodal representation of the building is utilized in the automatic generation of the network. Figure 5 illustrates graphically the steps used in the network design algorithm. The algorithm, which is recursive in nature, is outlined below:

Step 1: Find the maximum and minimum x and y coordinates of the terminals in a zone.

Step 2: Establish a cluster size (usually 9 to 10 m). If any of the distances are smaller than or equal to the cluster size, the zone has one cluster, else divide the zone into clusters.

Step 3: Generate branch along the longer axis of the cluster as centrally as possible. Connect the terminals to this branch. The center point of the cluster branch is the cluster point.

Step 4: Follow step 3 for all the clusters in the zone. Repeat Step 3 on the cluster points. This is the zone branch. The center point for the zone branch is the zone point.

Step 5: Connect each zone point to the start node, starting with the furthest zone point.

Step 6: The system node is similarly connected to the start nodes (this is not shown in the figure).

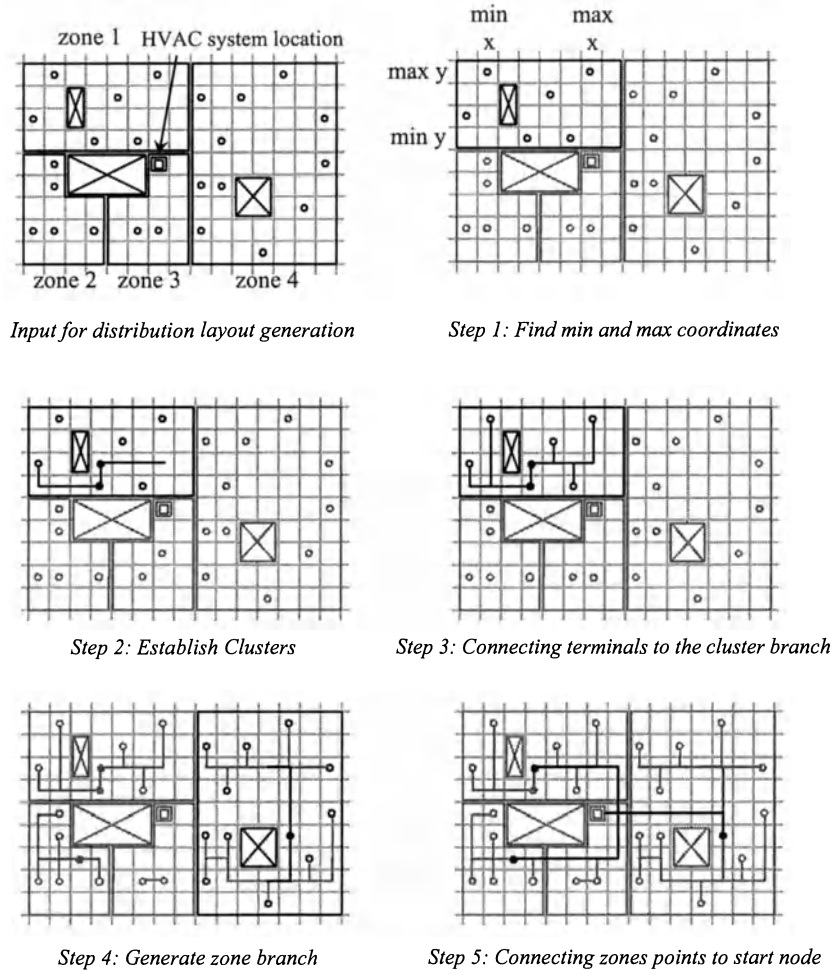


Figure 5. Distribution network design sequence

At any step, only two nodes are connected to each other using the shortest path algorithm (Horowitz et al. 1993). The distance used between any two nodes to calculate the shortest path considers the actual length of the path, any turns in the path, and whether it crosses any terminal or branch. The use of this algorithm, in conjunction with the node discretization of the building, ensures that obstructions and openings (voids) are taken into consideration while deciding the distribution network's configuration. The

algorithm can be applied for multiple systems serving a zone or the whole building.

Once the layout nodes are obtained, the design agent parses the node information to generate a list of actual distribution network and establishes the connections between them and the terminal and generation components. Examples for distribution network components are duct, pipe, fitting, valve, damper, VAV (variable air volume) box.

At this stage the DOM may be considered complete and the thermal performance analysis of the building can be conducted.

4. ILLUSTRATIVE EXAMPLES

4.1 Objective

To demonstrate the functionality of the system, we offer some illustrative examples of detailed performance analysis of complex buildings and their systems, conducted in early stages of design.

4.2 Duct layout generation and sizing

An actual office building (Building A), located in Pittsburgh was selected for the automatic layout generation case study. By choosing this building for case study, it is possible to compare the automatically generated layout with the layout independently designed by the project's mechanical engineer. Building A has two floors and is served by five Roof-Top Units (RTU). To demonstration the layout generation, only one floor has been simulated. Four of the RTU's serve the open plan office space. The distribution is floor-based and open-plenum with ducts extending from the vertical shaft to the VAV boxes. The fifth RTU serves the core area, is ceiling-based and with a ducted plenum.

This study involved the generation of duct layout given the location of VAV boxes. Here the location of the vertical shaft constitutes the start node and the location of the VAV boxes the terminal nodes. As Figure 6 and Figure 7 demonstrate, there is a good match between the actual network for the core and the open plan office area (as designed by the mechanical engineer) and the computationally generated network.

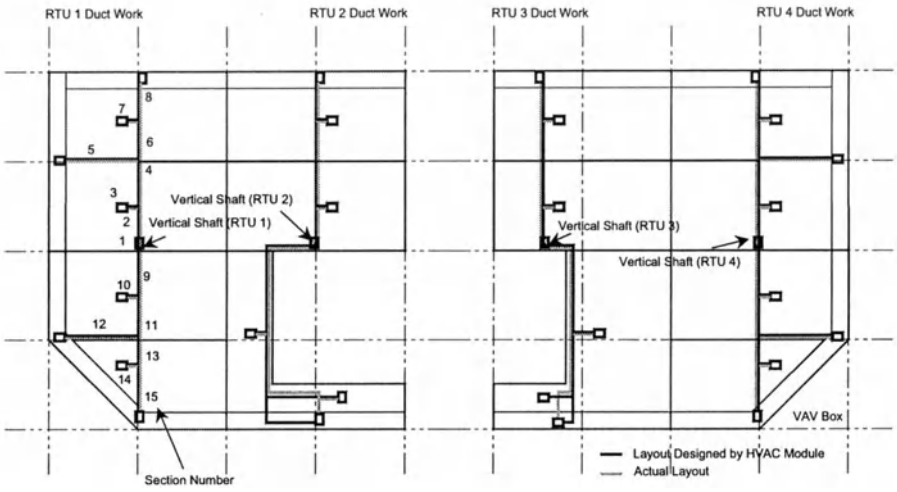


Figure 6. Automatic duct layout generation: open plan office area, Building A

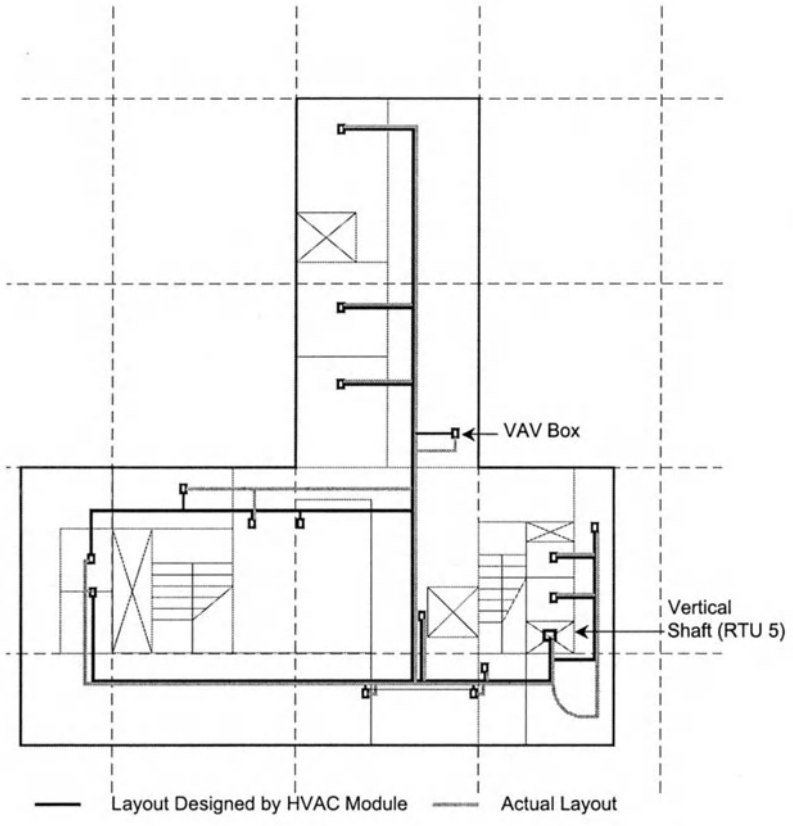


Figure 7. Automatic duct layout generation: core section, Building A

For RTU 1, the sizes of the ducts designed by the HVAC design agent are compared with the actual sizes in Table 1. The sizes designed by the agent are within ±20% of the actual sizes. This is generally considered to be acceptable range, as various sizing methods result, in practice, in variations in the ±50% range (Brahme 1999).

Table 1. Comparison of Duct Sizes for RTU1

Section number	Length (m)	Flow (l/s)	Actual duct diameter (m)	Generated duct diameter (m)	Percentage difference
1	1	4969	0.84	0.84	-0.7
2	5	2652	0.66	0.75	-13.1
3	2	812	0.43	0.41	2.8
4	5	1840	0.58	0.62	-6.5
5	9	566	0.41	0.34	16.9
6	4	1274	0.51	0.52	-0.6
7	2	708	0.41	0.38	5.2
8	5	566	0.38	0.33	13.7
9	5	2317	0.64	0.70	-10.3
10	2	812	0.43	0.41	2.7
11	5	1505	0.56	0.54	3.2
12	9	481	0.36	0.31	14.9
13	3	1024	0.48	0.44	9.0
14	2	543	0.38	0.31	18.9
15	6	481	0.36	0.30	16.0

4.3 Energy analysis of a air-based system

To demonstrate the systems capabilities for the computational analysis of the annual energy consumption and thermal indoor conditions, a typical office building located in Pittsburgh was selected (see Figure 8). The building consisted of five zones - one interior zone and four perimeter zones. The mechanical system was assumed to be of Constant Air Volume (CAV) type with reheat coils at the zones. The diffuser and thermostat locations were given as input information. Using its embedded design agent, the HVAC application generated the duct layout for this diffuser configuration and automatically sized it. Figure 9 shows the simulated temperature profiles for the cells in Zone-West for the month of January. Figure 10 shows the heating, cooling and the fan energy consumption for four months – January, April, July, and October.

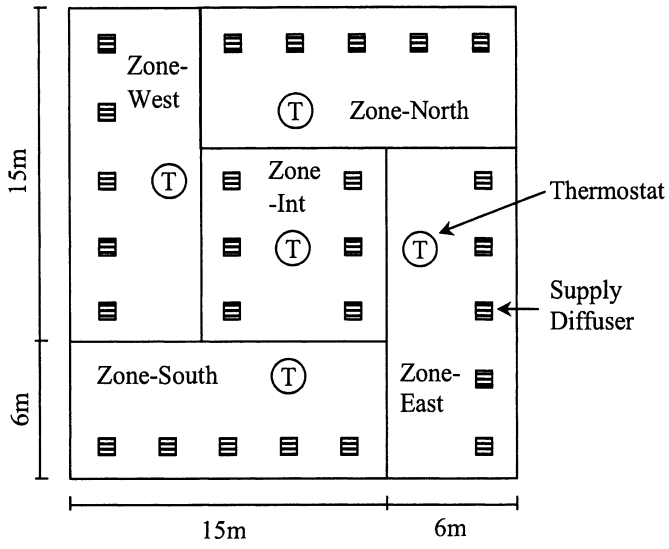


Figure 8. Typical office building plan

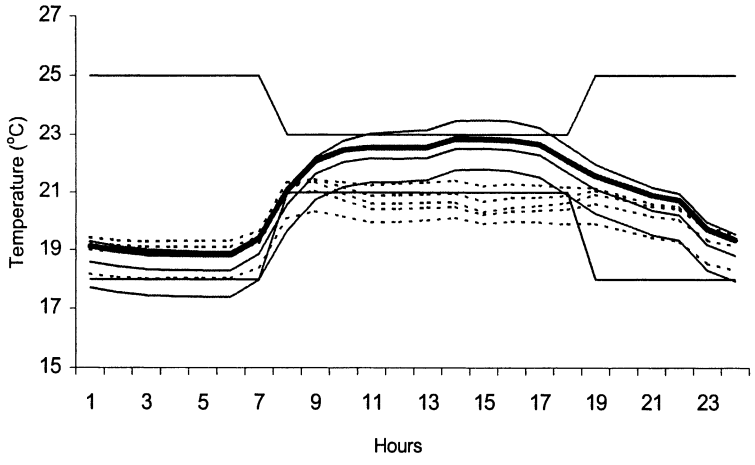


Figure 9. Temperature profile for Zone-West (January)

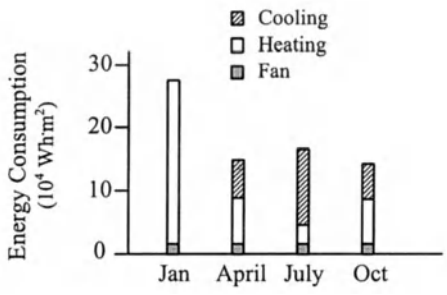


Figure 10. Energy consumption

5. CONCLUDING REMARK

We presented a computational approach to support an extended set of early-design performance queries based on a differential building representation and the use of generative agents for the design and modelling of buildings' technical systems. Using the domain of building energy systems as a case in point, we demonstrated that complex technical systems may be subjected to detailed performance analysis even in the early stages of design. We believe that by adapting this approach, detailed computational building performance assessment tools may become more accessible to a general user, extending its relevance beyond the realm of domain experts.

6. REFERENCES

Brahme, R., 1999, *Computational Support for Building Energy Systems Analysis;* Ph.D. Thesis, School of Architecture, Carnegie Mellon University, Pittsburgh, Pennsylvania.

Horowitz, E., S. Sahni, and S. A. Freed, 1993, *Fundamentals of data structures in C;* Computer Science Press, An imprint of W. H. Freeman and Company, New York.

Mahdavi, A., 1999, A comprehensive computational environment for performance based reasoning in building design and evaluation. *Automation in Construction 8.* pp. 427 – 435.

Mahdavi, A., E. Ilal, P. Mathew, R. Ries, and G. Suter, 1999, "Aspects of S2" *Proceedings, CAAD Futures '99*, Atlanta, Georgia.

Mahdavi, A., and N. H. Wong, 1998, "From Building Design Representations to Simulation Domain Representations: An Automated Mapping Solution for Complex Geometries" *Computing in Civil Engineering; Proceedings of the International Computing Congress, 1998 ASCE Annual Convention.*

Mahdavi, A., P. Mathew, N. H. Wong, 1997, "A Homology-based Mapping Approach to Concurrent Multi-domain Performance Evaluation" *Proceedings of The Second Conference on Computer Aided Architectural Design Research in Asia: CAADRIA '97.* Hsinchu, Taiwan. pp. 237 - 246.

Highlighting the affordances of designs

Mutual realities and vicarious environments

Christopher Tweed
School of Architecture, The Queen's University of Belfast

Key words: affordances, human-environment interaction, design evaluation, agents

Abstract: Computer-aided evaluation of predicted design performance is an enduring theme within CAAD research and practice. However, most evaluative systems address aspects of design that are readily amenable to formal or quantitative treatments. Analyses of how people use and interact with designs rarely progress beyond a narrow functionalism, in which 'the user' figures as a type with poorly defined needs and characteristics. This paper outlines a theory of actor-environment interaction based on Gibson's notion of affordance as a precursor to exploring how computers can be used to highlight the affordances of designs. Two simple prototypes are described. The main conclusion is that while computers are unlikely to be able to detect affordances, they can generate and present information in ways that will enable human designers to appreciate more fully the possible implications of their designs for a broader range of potential occupants.

1. INTRODUCTION

Computer-aided evaluation of predicted design performance is an enduring theme within CAAD research and practice. However, most evaluative systems address aspects of design that are readily amenable to numerical treatments, such as energy modelling, structural and cost analyses and lighting design. When CAAD *is* used to explore design solutions it is generally to visualise 3d form. Existing applications, therefore, prioritise quantifiable and visible aspects of buildings over others. Precisely because these types of CAAD tools are widespread, there is a tendency for designs to be assessed mainly, or only, against a limited set of parameters. This leads to a form of reductionism in which buildings are viewed primarily as formal

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 681–696.
© 2001 *Kluwer Academic Publishers.*

compositions, energy consumers, structural systems or 3d sculptural objects. But, as has been argued elsewhere (Tweed, 1997), the prioritisation of the visual, to the neglect of non-visible aspects of design, leads to an impoverished architecture. The most revered buildings are generally those which engage all of our senses, as well as our intellect. If CAAD is to make significant contributions to raising the quality of design then it will have to embrace a wider range of concerns, including the kind of experiences designs offer their eventual occupants.

Designers tend to treat the users of their designs in stereotypical fashion—the user is a single 'type' whose characteristics are rarely fleshed out in any detail. As a result, designs often ignore the special needs of many potential occupants, e.g. children, mothers, the elderly and disabled. Increasing emphasis on the need to provide inclusive environments, suggests we need to account for a much wider range of occupants than we do now. Hence, there is considerable advantage in developing CAAD tools which will allow designers to assess their designs from the viewpoints of different potential users or occupants.

The purpose of this paper is to suggest ways in which computers might be able to highlight the use-value of designs for a variety of different occupants. The research attends to their heterogeneous needs and how these might be satisfied or denied by a design. The intention is to use CAAD to explore fundamental human-environment interactions. The paper begins by outlining theories, emerging from ecological psychology and anthropology, about how people interact with their surroundings. The discussion then considers how these theories might be applied in a computational environment, illustrated by examples of prototype applications with different levels of computational ambition. Finally, the paper presents a discussion of issues surrounding the application of the theory of affordances to CAAD, in particular in the knowledge each type of system requires of the design and its possible occupants.

2. HUMAN-ENVIRONMENT RELATIONS

2.1 Functionalism in architecture and CAAD

Buildings are generally assumed to have well-defined functional properties which can be treated and assessed independently of actual users. The origins of this view can be traced back to the early part of the 20th century when functionalism emerged as a central plank of the modernist agenda. In functionalism, architecture is primarily a matter of

accommodating the functions that are prescribed for it. Architecture must work efficiently by conforming to ideal, scientifically generated and tested user requirements. Le Corbusier's definition of the house as a "machine for living in" and Louis Sullivan's maxim "form follows function" epitomise the functionalist approach. While specific architectural devices of functionalism are many and varied, they often share a preference for parsimonious design, with unadorned and clearly expressed structure and fabrication methods (Rowe, 1987). Spatial organisation is derived from a well-defined programme of uses and activities. Designers sought optimal spatial configurations for accommodating human activities. Strict hierarchies of uses and elements were contrived to categorise prominent and subordinate areas of activity. These were often ergonometrically determined and, because of the emphasis on machine efficiency, were usually planned to satisfy minimum space standards. Designing circulation within buildings was often reduced to the problem of maximising the proximity of use-functions to minimise travel distances. Planning of buildings centred on idealised descriptions of the occupants and their activities, such as eating, sleeping, cooking, washing, and bathing. Spaces dedicated to these activities were designed as distinct (and often, separate) activity areas.

In the second half of the last century it became clear that the prevailing theories of function could not account for the richness of everyday interactions between people and their designed environments. So, whilst functionalism had, and still has, a core intention of meeting users' practical requirements, the notion of 'the user,' and his or her interaction with actual buildings was narrow. Plying a strongly behaviourist line, functionalism tends to treat occupants of its buildings as simple biological organisms with highly predictable behaviour to match.

Many contemporary designers still adhere to some form of functionalism. It is not surprising, therefore, to discover that recently completed empirical studies of work environments show that the designs of even greatly admired, contemporary architects often ignore how people interact with their surroundings (Koffka, 2000). CAAD, with its penchant for "rational" (functionally systematic) methods of design is both strongly influenced by functionalism and furthers its application.

2.2 Direct perception and affordances

In *The Ecological Approach to Visual Perception*, J.J. Gibson (1979) argues that our understanding of the world, as acquired through perception, is the result of our continuing engagement with the world. Opposing the prevailing cognitivist orthodoxy, Gibson claims that perception is not a matter of a mind processing the 'raw data' supplied by the body's sensory

apparatus, but is a single act in which the human being (body and mind) directly perceives its environment. Perception is not, therefore, a two-stage process of sensing followed by interpretation, but is the direct apprehension of what an environment offers, or *affords*. The pairing of an organism with an environment results in a more or less unique set of *affordances*. In general terms, an affordance is what an environment offers an organism, for good or for ill. Thus, we can say that trees afford climbing for squirrels, but not for dogs. An affordance only emerges through the pairing of an organism and an environment. Thus, "[a]n affordance is neither an objective property nor a subjective property. It is both. An affordance cuts across the subjective-objective dichotomy and in doing so highlights the inadequacy of this dualistic thinking. It is equally a fact of the environment and a fact of nature. It is both physical and psychical, yet neither. An affordance points both ways, to the environment and to the observer." (Gibson, 1979, 129).

Affordances, therefore, are the results of, and emerge from, interactions between people and their environments. For a given environment, they differ depending on the individual, his or her circumstances and the characteristics of the general situation; they recognise the uniqueness of human interaction rather than presupposing objective functions for an environment.

Gibson is critical of the fact that a theory of affordances is lacking from architecture: "...a glass wall affords seeing through but not walking through, whereas a cloth curtain affords going through but not seeing through. Architects and designers know such facts, but they lack a theory of affordances to encompass them in a system." (Gibson, 1979, 137).

Unfortunately Gibson provides little more than a sketch of what a theory of affordances might look like. More recently, Ingold (1992) has begun to flesh out Gibson's ideas in greater detail, and introduces the term *effectivities* to denote the physical and psychological properties of individuals. Effectivities define what an organism is capable of doing—for example, in physiological terms, its mobility—in a given environment. Effectivities of organisms and properties of different environments combine to define what an environment affords a given organism. For Ingold, humans are like any other organism, with one important difference: people design. Humans are the authors of a large part of their environments, as well as being 'users.' Ingold, however, is adamant that 'culture' has little or no place in accounting for which affordances 'show up' for different organisms.

Dreyfus (1996), however, emphasises the cultural component with reference to the philosophy of Merleau-Ponty (1962). For him, an affordance is seen as a culturally conditioned response to an artefact, such as a chair. Dreyfus argues that a chair affords sitting on because we have the kind of bodies that bend at the back of the knee and because sitting is a culturally defined norm in many situations. Many, rather than all, because there are

occasions when sitting in a chair may not be culturally or socially sanctioned—in pre-westernised Japan, or in the presence of certain others with different social standing.

As Dreyfus's account suggests, we might also need to include culturally defined skills as one of what Ingold calls an organism's effectivities: "J.J. Gibson, like Merleau-Ponty, sees that characteristics of the human world, e.g. what affords walking on, squeezing through, reaching, etc. are correlative with our bodily capacities and acquired skills, but he then goes on ... to add that mail boxes afford mailing letters. ... Affords-mailing-letters is clearly not a cross-cultural phenomenon based solely on body structure, nor a body structure plus a skill all normal human beings acquire. It is an affordance that comes from experience with mail boxes and the acquisition of letter-mailing skills. The cultural world is thus also correlative with our body; this time with our acquired cultural skills." (Dreyfus, 1996).

2.3 Affordances and design

Within design, we find some ideas which come close to capturing the notion of affordance. As part of the 'populist' reaction against extreme functionalism, Alexander's 'pattern language' attempts to understand the relation between bodily capabilities and preferences, cultural settings and designs (Alexander et al, 1977). Alexander's work is revered and reviled in equal measures by the architectural community. The almost complete subordination of classical formalisms to practical and symbolic concerns has alienated many architects. Sadly, the underlying concern with offering enriching and interesting experiences for the occupants—which is at the heart of *A Pattern Language*—is thrown out along with the populist approach. For this research, the most important contribution of Alexander's work is the way in which it points up issues of occupancy which are rarely, if ever, discussed, such as the need for shelving at a certain height, or the social interactions which different degrees of enclosure permit and encourage. Ultimately, however, the patterns are too prescriptive and rely on too much uniformity of purpose, need and desire in the imagined occupants.

The lack of attention given to aesthetic and formal considerations in Alexander's work may be remedied by absorbing aesthetics within an extended concept of affordance. There is no reason why we cannot speak of a building affording delight, as well as commodity and firmness. Papanek (1985), for example, broadened the notion of function to embrace aesthetics in what he called the 'function complex.' His aim was to dissolve the harsh distinction between utility and aesthetic pleasure. He quotes his students as asking "'Should I design it to be functional ... or to be aesthetically pleasing?' ... 'Do you want it to look good, or to work?'" (Papanek, 1985,

19). Whether we can ever successfully embody this way of thinking within CAAD remains to be seen.

3. EFFECTIVITIES

Effectivities are those characteristics of individuals that decide which affordances emerge from interactions between occupants and environments. *Table 1* lists some of effectivities required of an actor and the corresponding properties of a door needed to confirm the affordance of passage. Most design concentrates on physical effectivities, for which standard ergonomic and anthropometric design guides are available. The extent to which these apply will depend on how closely the occupants conform to statistically defined norms.

Table 1. Some effectivities required for the affordance of passing through a door.

Category	Issue	Effectivity	Door property
Physical	Is the occupant physically capable of passing through the door?	Dimensions; mobility; dexterity; physical strength needed to open and hold door.	Dimensions of door and approach; operability of door equipment.
Psychological	Is the actor pre-disposed to pass through the door?	Personality; mood; need of passage; knowledge of, and attitude towards, what lies on the other side of the door.	Symbolism.
Social	Is it socially acceptable for this actor to open/pass through the door?	Social or organisational standing; right to use door and connected space.	Symbolism and signage; locking.
Cultural	Is the actor aware of the practices surrounding the use of doors?	Familiarity with door-using culture.	Transparency of operation.

In this broad view, it would be necessary for all of these effectivities to be positive for the affordance of passage to emerge from a specific actor-environment interaction. And the list is not exhaustive. Even for this very simple example, the array of effectivities which needs to be considered to confirm or deny the affordance is huge. So, the example alerts us to the complexity of interactions and to the difficulty in identifying affordances in designs. We shall return to this topic when we consider the computability of affordance recognition.

Specialised areas of design offer detailed consideration of effectivities, particularly where they are considered to depart from societal norms, such as

for those with physical or cognitive disabilities. In gerontology, terms such as functional performance and competency are used to describe how well an individual is able to carry out everyday activities. Design guidance for frail elderly emphasises the need to consider actor-environment interaction in much greater detail (Schwarz and Brent, 1999).

3.1 When is a door not a door?

We conclude the first part of the paper by considering a common feature of all buildings—the door—and the affordances it offers different actors.

At the most abstract level a door provides a variable opening between two spaces. To afford passage, a door must open. When closed, the door will be expected to offer varying degrees of resistance to flows between either space. A closed door between outside and inside would normally be expected to prevent people from passing in either direction and stop the penetration of wind and rain from outside to inside. Less predictably, the door may or may not allow light to pass in either direction. Door glazing, therefore, although common, is not necessarily a universal feature of doors. Not all doors afford seeing-through. Even when glazing is used, the amount and type may vary enormously.

When open, a door will present an opening with shape and dimensions which will determine what kind of objects can pass through it. With most doors, the opening can vary from a thin vertical strip to wide open. But perhaps just as important, the immediate environment of the door will also determine what and how things can pass through its opening. Numerous comic situations have been created by the difficulty in moving furniture through doorways. To reveal a passage, the door must move and take up a new position. Fledgling designers are mostly aware of the opening behaviour of doors, certainly of those that open conventionally, by swinging on hinges, but still forget that doors are dynamic objects and that opening is a process. Many forget that a sliding door has to slide somewhere on a track and that, with the exception of concertina and folding varieties, the shape and size of the door will remain constant throughout the opening process.

Moving to a lower level of analysis, we turn our attention to the way in which a door is opened. To ensure that the door stays closed when required to do so, most will have a latch which engages with a striking plate on the frame. To open the door requires that we retract the latch from the plate by activating a lever or turning a knob. The obstacles to retracting the lever are potentially numerous: the handle is missing or falls off; the lever mechanism of the hinges have seized because of corrosion; the door is locked; or the handle cannot be reached. This last obstacle raises the question of how, and for whom, a handle becomes inaccessible. A handle placed 1 metre above

floor level will be accessible to most ambulant adults, but not necessarily to a 3-year old child.

In summary, a given door affords passage for a range of different people, organisms, and things; a potential barrier to precipitation, moisture, light, sound, gaze, air movement; and, with a suitable locking mechanism, control over the movement of people and things between the connected spaces. It should be noted that it is impossible to exhaustively identify the affordances of any door, because new affordances arise from encounters with new actors.

4. BEING IN THE DESIGN

The large number and sheer complexity of factors which need to be considered to grasp the possible interactions between different occupants and a future building makes the identification of affordances a daunting task. It poses some interesting research questions: what is the most fruitful way for us to understand what a design offers different groups of users? Can computers assist in this task? And, if so, how?

Computers are already able to tell us (more or less accurately) how much a design will cost, how much energy it will use, what sort of environment it will offer and what it will look like from an infinite range of viewpoints. These capabilities, in the broadest sense, can inform the identification of affordances. We can legitimately ask, for example, does a building afford energy conservation? In this paper, however, the crucial question is can we extend our analyses to gain some idea of what it will be like to *inhabit* the design?

It should be obvious from the above that the number of potential affordances in a design is very large, and quite possibly infinite. To attempt to identify everything (for good or ill) a design affords is neither possible nor desirable. It will be necessary to highlight those which seem most important for different actors.

We begin by examining how existing methods of simulating experience measure up to the theories described above.

4.1 Activity modelling

Most activity modelling is concerned with the movement of people around buildings, for example, in the event of a fire or other emergency. Activity modelling, therefore, often implies a particular theory of how people behave in buildings and of their practical needs. A crucial difference between activity modelling and the approach adopted by this research is the distinction between activities and practices.

The activity of sitting spawns many different practices, depending on the cultural and social context. Washing oneself is an activity with reasonably well-defined spatial requirements, for different configurations and types of washing devices (showers, baths, lavatory basins), but bathing, as a set of practices, admits other concerns, such as degree of privacy (acoustic as well as visual), the implied connections to drying oneself and the nearness of drying towels etc. Anthropometric data on heights for shower outlets, shower trays, baths, and lavatory basins are necessary, but not sufficient, to describe the practices of bathing, washing and drying.

To return to our earlier example, if we consider the activities doors support, in isolation from the effectivities of those who interact with doors, we exclude the practices that grow up around people and doors. The difference is best illustrated between the way in which people expect things to be used—their 'proper' function—and the way in which they are actually used, by different people. Fire doors are a good example. When is a fire door not a fire door? Precisely when it is wedged open by the occupants of the building. It is this gap between intended function and actual use that we are trying to address in this research.

4.2 Virtual environments

It is entirely feasible that in the future it may be possible to configure virtual environments to simulate the effectivities of different users, and there is already some progress on this front, for example, in a test rig developed at Strathclyde to simulate movement through a building in a virtual wheelchair. But until problems with nausea and perceptual realism are solved, or systems become less expensive, virtual worlds will remain highly specialised, beyond the reach of most practising designers. Rather than virtual realities or environments, we need design tools which support the development of mutual realities and provide vicarious environments through which designers can gain a deeper understanding of a range of viewpoints on their designs.

5. HIGHLIGHTING THE AFFORDANCES OF DESIGNS

The remainder of the paper describes work carried out by the author on two prototypes which have the broad aim of highlighting issues surrounding the inhabiting of designs.

5.1 The "Walkies" demonstrator

The 'crit' is a staple of architectural education, during which instructors question students about all aspects of their design, including its suitability for occupation by different people. Many of the questions are about what it would be like for a user to experience the eventual building. To promote a greater sense of vicarious involvement with designs and to direct attention to specific problems, we developed a simple prototype display system which allows representations of different users to be moved around plan drawings. This was largely an attempt to animate the diagrams produced in paper design guides.

The 'situated sprite' is the most basic technique for directing attention to possible affordances of designs. The sprite is a graphical representation of different types of occupant. A range of sprites has been designed for use on plan drawings, as plan views of the types of occupant they represent. One sprite, for example, is a plan view of a person in a wheelchair; another is of a person pushing a child's 'buggy.'

A demonstrator was constructed using Macromedia *Director* which allows the user to import a plan drawing as a backdrop to the *Director* 'stage.' During a crit, an instructor can 'pick up' an appropriate sprite and use this to 'walk' around the plan—hence the system moniker of "Walkies." This very simple system provides little sophistication, and does little other than automatically orient the actor to the direction of cursor movement. *Figure 1* shows a series of screenshots showing the movement of an actor around a furniture arrangement. The main aim in developing this system was to provide a focus for dialogue between designers and potential occupants of the resulting design (Tweed and Woolley, 1992).

The system's knowledge of the design is minimal. It requires only the ability to import a bitmap at a given scale. The system is flexible, but limited. It relies on its users to make the links between the represented actor and the depicted design.

This system, though extremely easy to implement, is useful in drawing attention to potential problems in designs merely by suggesting the inhabiting of a design by different actors. We must be careful not to complicate design tools simply because they do not use the latest or most sophisticated technologies. However, it is clear from these early experiments that there is further potential in this approach.

Whilst technically trivial, the utility of merely drawing attention to the presence of different types of users in a design should not be underestimated. Anthropometrically accurate sprites, used consistently during design tutorials, serve as a focus for the ensuing discussion surrounding the experience of a design by different user groups.

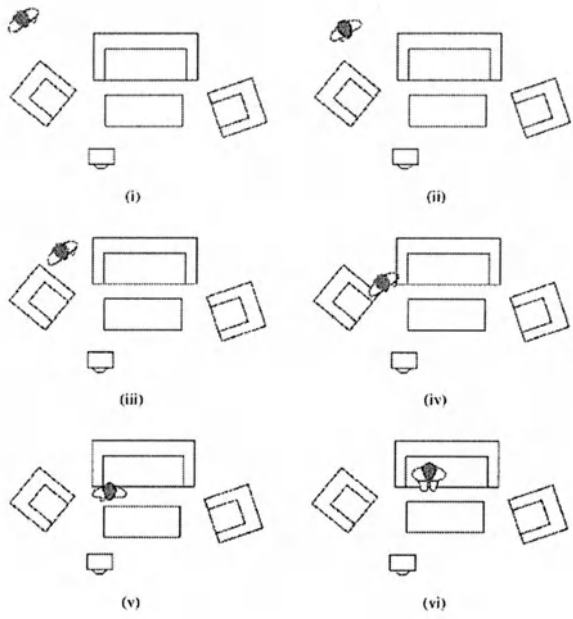


Figure 1. An early exploration of movement in a space.

Perhaps it is sufficient for computers to merely remind designers that different types of user are likely to use their buildings. Rather than aim for a highly ambitious and complex automated affordance evaluator, it may be just as fruitful to suggest issues that need to be addressed *by users* to evaluate a design proposal.

6. THE *ADEPT* SYSTEM

Although a very modest step, the "walkies" program has inspired us to seek further ways of conveying the latent possibilities of experience in a design for different actors. This is an ambitious aim, which will take a long time to realise. It is a programme of research rather than a single project. We are developing the *ADEPT* (Affordances of Designs Evaluation and Presentation Toolset) system as an expandable set of computer-based tools which will highlight different aspects of a design and so allow designers to view their efforts from a range of different concerns. It consists, at present, of a series of exploratory studies rather than useable prototypes. Two types of study are being investigated. The first addresses accessibility; whereas the second examines ways of displaying dynamic sensory fields centred on different types of actor.

6.1 Development environment

The development environment for both consists of the *ArchiCAD 6.0* drafting, modelling and rendering system, the Applications Programming Interface (API) for *ArchiCAD*, and the *CodeWarrior* Integrated Development Environment (IDE). *ArchiCAD* is a comprehensive architectural modeller which offers the user a range of parameterised building elements with which to describe their designs. In addition to conventional elements such as walls, windows, doors, roofs, and slabs, the system allows users to define their own parameterised objects as library parts. *ArchiCAD* comes with its own library and there are also several extensive third party libraries available. 2d graphical primitives are defined too. As is now commonplace in most CAD systems, *ArchiCAD* provides tools and commands to create, define and manage model layers, element materials and their properties, 3d views, and animations. One of the most useful additions to recent versions is the zone. Zones are primarily 2d area fills which can be manually or automatically defined. A zone, however, can have non-graphical information attached to it.

The *ArchiCAD* API allows the programmer to interrogate and change the element and attributes databases via 'add-on' modules. These add-on modules are accessed through items added to *ArchiCAD's* menus and tool palettes. Add-ons can modify existing or create new elements and attributes (e.g. layers). The programmer can define handlers for a set of events generated by *ArchiCAD*.

For our purposes, the only major drawback is that with the current API there is no mechanism for intercepting low-level events such as mouse clicks or information such as the current position of the cursor. Add-ons can only be activated by specific user requests. This prevents us from providing a display within *ArchiCAD* which is automatically updated in real-time as a sprite is moved around the plan. We are considering two different solutions to this problem. First, it may be necessary for add-on modules to provide and control their own windows to display information. This would require the add-on module to replicate the graphical functions which the CAAD system already provides. The other, less attractive option is to abandon direct interaction with *ArchiCAD* and export models to a dedicated application.

6.2 Occupancy and accessibility

Perhaps the simplest affordance we can address using CAAD is accessibility. We have developed an exploratory prototype to examine ways in which the accessibility of a design can be assessed quickly and easily. In its present form, the prototype analyses the geometry of a plan and identifies

the obstacles it presents for different users. To do this, potential users must be represented in the CAAD system with the information needed to determine which parts of the design represent an obstacle.

As shown in *Figure 2*, the designer places a potential actor (chosen from a range of available library parts) on the plan of the building, selects this actor and then, via a menu command, asks *ADEPT* to show accessibility for that actor from that point. The system extracts from the actor's description basic information, such as the width of the actor and compares this effectivity with properties of the design.

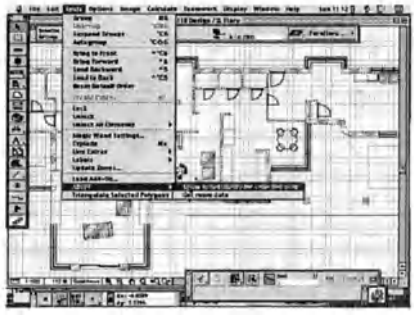


Figure 2. the accessibility menu command.

Figure 3 shows the resulting accessibility zone for the selected actor superimposed on the plan view. For now, the accessibility function takes no account of furniture arrangements and assumes that an actor can move freely within the spaces indicated. Accessibility is determined solely from the actor's width and available door openings.

Figure 3. the resulting accessibility zone.

This relatively simple example raises important points for discussion. If it is known in advance that this house will be occupied by a person in a wheelchair then much of the design discussion will probably centre on

relevant issues, such as accessibility. But if, as in this case, there has been no mention of mobility then tools such as this could operate as a simple check, requiring little effort on the part of the designer, but highlighting some of the problems which might occur.

6.3 Describing dynamic perceptual environments

Many basic affordances depend on the changing physical environmental conditions in a space and are tied to thermal, visual and auditory comfort. We have begun to examine integrating dynamic thermal, auditory and visual analyses with *ArchiCAD* to provide real-time display of sensory fields surrounding an actor. This work is in its early stages. *Figure 4* shows a mock-up of the kind of display from this type of analysis. For now, this is limited to displaying the visual field for a given actor with a given orientation. The intention is that several sensory fields will be displayed simultaneously, so that a designer can see the intersections and overlaps between lighting, thermal and acoustic environments. For the thermal environment, it is proposed to use the author's existing thermal modelling package to show the conditions surrounding the actor, including the radiative components of heat exchange between the actor and room surfaces. Convective conditions pose greater problems as they will require the integration of a computational fluid dynamics (CFD) package. However, if calculations are carried out on a room-by-room basis this should be possible within real-time.

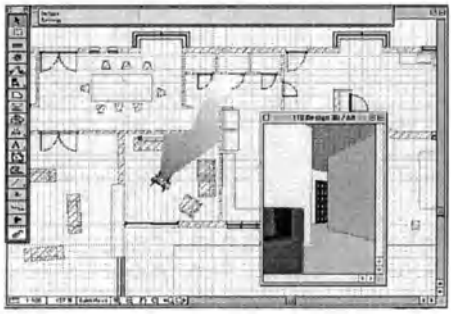


Figure 4. highlighting the visual field.

6.4 Discussion

The *Walkies* demonstrator represents a low-ambition, low-functionality, highly flexible critiquing system. It 'knows' nothing about the design or occupants it displays. It requires that its operators do most of the work. This

is CAAD operating primarily as a medium rather than as a tool. In contrast, the *ADEPT* system requires knowledge about the building in a particular form (as an *ArchiCAD* model.) In both cases, the salient point is that CAAD systems always exist in some context, and a significant part of their context is social (Tweed, 1998).

The principal advantage of *ADEPT* is that it draws attention to important features of designs which would otherwise remain tacit, and beyond critique. The prototypes are at a very early stage of development and there are technical and conceptual issues to be overcome to progress further. Specifically, any form of automated affordance detection seems highly unlikely. However, even the limited experience in using these prototypes suggests that we can do something useful with CAAD to broaden its scope and thereby address wider issues in design.

7. CONCLUSIONS

The highlighting of affordances is likely to remain a strictly human capability, but computers can assist designers by presenting relevant information about the environment and its inhabitants in ways that make it easier to consider their interactions. The task of highlighting affordances, therefore, is not one that requires sophisticated algorithmic treatment, but rather one of information design. Simply presenting information in more appropriate forms and contexts, as an integral part of the CAAD system—and at the right point in the design process—can help to direct attention to possible problems (and delights) for different occupants. In deciding where to allocate the responsibility for affordance detection, we need to consider CAAD systems and their operators (designers) as a whole system. Just as this paper has argued for the need to consider buildings in relation to those who will use them, so the same logic applies to CAAD systems and their users. A major part of our future research is to identify the appropriate division of labour between machine and human.

8. ACKNOWLEDGEMENTS

I am indebted to many people who have helped to shape the ideas presented in this paper. In particular, I would like to thank graduate students in the Department of Architecture, Carnegie Mellon University, Pittsburgh—Jason Brooks, Brian Gardner, Can Gunduz and Smita Gupta—with whom I enjoyed many fruitful discussions about the computability of affordances. I

would also like to thank *Graphisoft* for providing the API for *ArchiCAD* free of charge.

9. REFERENCES

Alexander, C., Ishikawa, S. and Silverstein, M., 1977, *A Pattern Language: Towns Buildings Construction*, Oxford University Press, New York.

Altman, I., Lawton, M.P. and Wohlwill, J.F. (eds.), 1984, "Elderly People and the Environment", in: *Human Behavior and Environment: Advances in Theory and Research*, Volume 7, Plenum Press, New York.

Dreyfus, H.L., 1996, "The Current Relevance of Merleau-Ponty's Phenomenology of Embodiment", in: *Electronic Journal of Analytic Philosophy*, University of Indiana, http://www.phil.indiana.edu/ejap/1996.spring/dreyfus.1996.spring.html, pp.8.

Gibson, J.J., 1979, *The Ecological Approach to Visual Perception*, Houghton Mifflin, Boston.

Koffka, B., 2000, "From caves to space stations." Presentation delivered in *The Robert L. Preger Intelligent Workplace*, Department of Architecture, Carnegie Mellon University, Pittsburgh.

Ingold, T., 1992, "Culture and the perception of the environment", in: E. Croll and D. Parkin (eds.), *Bush Base: Forest Farm – Culture, Environment and Development*, Routledge, London, pp.39-56.

Merleau-Ponty, M., 1962, *Phenomenology of Perception*, trans. from the French by Colin Smith, Routledge and Kegan Paul Ltd, London.

Papanek, V., 1985, *Design for the Real World*, Thames and Hudson, London, 2nd edition.

Rowe, P.G., 1987, *Design Thinking*, MIT Press, Cambridge, Massachusetts.

Schwarz, B. and Brent, R. (eds.), 1999, *Aging, Autonomy and Architecture: Advances in Assisted Living*, The John Hopkins University Press, Baltimore.

Tweed, C., 1999, "Prescribing designs", in: A. Brown, M. Knight and P. Berridge (eds.) *Architectural Computing from Turing to 2000: Proceedings of the 17th Conference on Education in Computer Aided Architectural Design in Europe*, eCAADe and The University of Liverpool, Liverpool, pp.51-57.

Tweed, C., 1998, "The social context of CAAD in practice", in: C. Branki and K. Zreik (eds.), *Cyber Design: Proceedings of the Seventh International Conference on Applications/Implications of Computer Networking in Architecture, Construction, Design, Civil Engineering and Urban Planning*, EuropIA Productions, Paris. pp.177-194.

Tweed, C., 1997, "The predominance of the visual in computer-aided architectural design", in: A. Asanowicz, and A. Jakimowicz (eds.) *CAAD - Towards New Design Conventions*, Technical University of Bialystok, Bialystok, Poland, pp.269-285.

Architectural critique through digital scenario-building

Augmenting Architectural Criticism and Narrative

André G.P. Brown
School of Architecture and Building Engineering, The University of Liverpool, UK

Key words: Digital recreation, Scenario-building, Narrative, Fake, Architectural critique

Abstract: As an idea *scenario-building* has parallels the use of creative faking in related disciplines, most particularly, in contemporary art. The techniques involved in scenario-building and faking offer us enhanced ways of undertaking creative thinking and critical review of architecture and architectural projects. Critical review and theoretical analysis of architecture can be undertaken via a range of methods that Attoe (1978) classifies as Normative, Interpretive and Descriptive. Digital representation now offers us new ways of augmenting these critical styles in ways that have yet to be fully exploited, and possible means of exploitation are illustrated in this paper. In short the work described here shows how digital techniques can be used to enrich architectural investigation, critical reporting and debate.

1. INTRODUCTION

The idea of the *scenario-building* is founded on principles similar to those used in creative faking in other related disciplines, but in particular contemporary art (Jones, 1992; Capaldi, 1979). The technique described is used as a device for creative thinking which can begin, for instance, with 'what-if?' scenarios. In this respect the start point is similar to the kind of *digital-unbuilt* work reported by Novitski (1998, 1999) and earlier by others (e.g. Cambell, 1995). The technique of creating, in a digital environment, works of architecture that have either not been constructed, or have been destroyed, offers a potential for virtual architectural environments to be developed in such a way that they can be used for more than simple appraisal. The visualisation, modelling and animation of works of

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 697–709.
© 2001 *Kluwer Academic Publishers.*

architecture can offer another set of media that can form the basis for a more rigorous and complex critical review. As Attoe (1978) says "the printed word is too limited to provide for all perspectives and nuances pertinent to the discussion of what is seen and experienced as three dimensional". Digital media can be the new ingredients in the critical review of, and response to, architecture.

In scenario-building the digital creation of what no longer exists, what never was, or what might be, is simply a first step. With a range of computer mediated techniques applied, the digital-unbuilt model can, for instance, form the start of a trajectory along a postulated historical path and associated circumstances. The idea of creating 'scenes that might have been' and using them as the basis for debate about possible plausible alternatives is referred to by Mitchell (1998) as creating a counterfactual argument. To some extent this paper is concerned with the creation of counterfactual arguments; but Mitchell notes that some historians regard counterfactual premises as meaningless. The contention presented in this paper is that such techniques can usefully augment architectural research and enrich architectural teaching.

As mentioned above digital scenario-building can be compared with, faking. The word fake is chosen deliberately to distinguish the associated ideas from those that might be associated with making a copy or facsimilie, which should be more correctly, referred to as a forgery. In forgery the perpetrator's aim is to copy an object and make the pretence that it is the real thing. Although the words forgery and faking are sometimes interchanged, here faking is taken to be the act of producing an alternative version of an object, or a set of facts surrounding that object to provoke debate and reaction. It refers, in this paper, to the idea that has been developed in the fields of art and logic where the proposition of alternative truth is used to foster richer debates, and to challenge accepted premises.

In this paper two specific applications are described to illustrate possible applications of scenario-building, and these are categorised by their critical styles. Critical styles rarely exist in isolation; a review or analysis is usually an amalgam of different critical approaches. The first of the critical approaches illustrated relies, principally on the use of Interpretive Criticism; particularly Advocatory and Evocative approaches. The example taken is that of investigating the potential significance of a particular building, and its designers, had a different set of circumstances prevailed in the 1920s and 1930s. In the second example Evocative criticism is again used, but a strong element of Self-criticism also prevails (Attoe, 1978). In this case, the technique is applied in a contemporary educational setting, and shows how students of architecture can usefully apply the scenario-building method to

their own work. The technique has met with praise and commendation (Hyett, 2000).

If we take Attoe's (1978) division of critical methods it could be said that in Computer Aided Architectural Design, the main attention, so far, has been on Normative Criticism. This takes as its premise that "somewhere in the world outside a building or urban setting there is a model, pattern, standard or principle against which its quality or success may be assessed". Techniques that use Artificial Intelligence, Expert Systems and Design Decision Support Systems would fall into this category. The aim of this paper is to support the potential of computer based systems to aid Interpretive and Descriptive methods of architectural criticism, to augment the methods used for Normative Criticism.

2. BEYOND DIGITAL UNBUILT

The application of digital modelling and consequent photomontage has become well established in architecture. This method has found application in examples such as new proposals for a site, or for the recreation of destroyed or unbuilt architectural projects. As a way of allowing us to visualise what does not exist, examine spatial qualities and engage in contextual criticism [a form of descriptive criticism according to Attoe (1978)] this has been very valuable. But we should note that it is clear that the nature of the digital representation in such cases fundamentally affects our perception of the value and quality of the architecture represented (Mitchell, 1992: Brown and Nahab 1996). This is a matter which is not dealt with in depth here, but is worthy of further consideration in the context of scenario-building.

Digital photomontage as a technique is a static one and based on two dimensional images. Over the past five years or so more computing capacity and better software has meant that we can contemplate the use of interactive three dimensional VR techniques to advance the systems for architectural design; but it is possible to use digital and associated VR techniques for more than simply visualising what once was, or might be desired. We can take the idea that what can be produced digitally can form the basis for more involved architectural critiques. The basis of the approach is that different scenarios (not just scenes) can be recreated digitally as a way of suggesting alternative versions of the truth, or alternative chains of development had the circumstances changed from those that prevailed. This then allows certain lines of argument to be supported: as such the technique can be used to augment critical theories of architecture (Brown and Simpson, 1997).

Figure 1. A Photomontaged Urban Scene

In short what is being proposed is constructive faking: constructive from the point of view that the creation of a faked set of events allows different lines of arguments and critique to be constructed. This is in line with the suggestion by Jones (1992) that the study of fakes should be embraced, and should be made with *"a greater awareness of the contingent and culturally conditioned nature of the distinctions made and the criteria applied"*.

3. EXAMPLE ONE: AS A POSTULATORY TOOL

To illustrate the scenario-building technique two examples are presented in this paper. The first example, here, takes a significant architectural practice, Connell, Ward and Lucas (Thistlewood and Heeley, 2001), that was active in Britain in the 1920s and 30s. Through the application of scenario-building interesting questions can be posed and the work of the practice can be examined. In its broadest sense, we can ask, for instance, whether the practice deserves a more prominent place in the history of the Modern Movement. Some of the critical reviews of the work of Connell, Ward and Lucas have been disparaging, perhaps because the work and the context of that work were often misunderstood and misinterpreted. The abandonment of what could be the practice's largest and most ambitious project, the Lord's Court, may, to some extent, have exacerbated the misunderstanding. This scheme is, therefore, ideal as a vehicle for illustrating the idea of using scenario-building, as a way of supporting a postulated idea.

Connell, Ward and Lucas adopted an approach to design that was radical, compared to most of their British counterparts. This radicalism covered a spectrum of design issues from colour treatment through to technological innovation. Through the use of virtual recreation of what would have been a

key project we can create a scenario to investigate their role in developing ideas relating to the Modern Movement. Through the scenario the effect of their potentially very potent influence on British and International architecture can be contemplated.

Lord's Court, dating from around 1929, was project that was started but was stopped at ground level (or so it was thought) because of wartime restrictions. It was never actually completed.

3.1 The source information

In practical terms the ideas being promoted in this paper are illustrated with a particular example, based initially on creating a virtual model. This model then acts as the core tool for historical and theoretical evaluation. The source data in terms of drawings on Lords Court plus information on projects prior to Lord's Court were provided by Heeley (1995). The data was not complete and what was found was difficult to trace. A typical original drawing, an elevation, is shown in Figure 2.

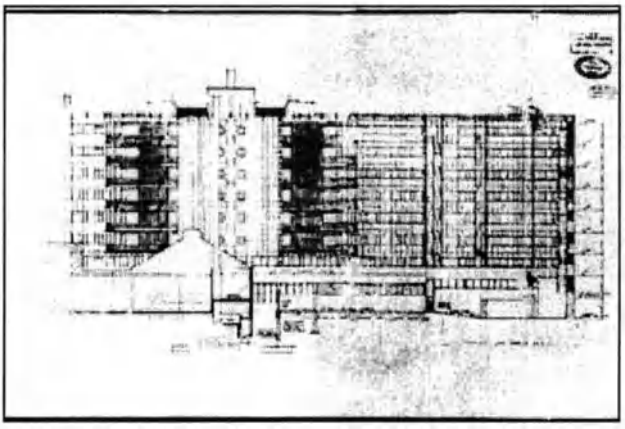


Figure 2. Original Connell Ward and Lucas drawing

The drawings and associated information were, until recently, thought to have been lost. The data that has been uncovered required careful study and cross checking to arrive at a point where the digital model could be constructed.

3.2 The digital representation

To create the models and visualisations the technique is very orthodox; AutoCAD and 3DStudio were used to generate the two and three-dimensional representations that can be seen in Figures 3 and 1. Some modification took place in Photoshop.

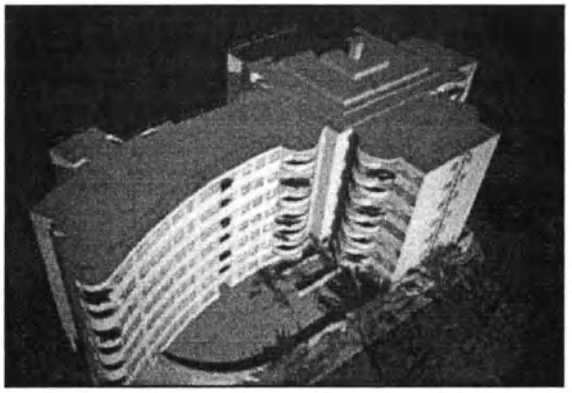


Figure 3. The building model

The drawings were in a poor condition and information was incomplete so some interpretation and extrapolation from other works by the same architects had to take place. For the architectural scholar this is where the interest starts to arise. It now becomes necessary to understand the context of the project in three respects. First, the prior and parallel work undertaken by Connell Ward and Lucas has to be understood so that the drawings can be read with a knowledge of the kind of technological and design innovations that the practice was keen to promote. Secondly, the contemporary architectural climate needs consideration. Thirdly the urban setting, the physical context at that time, has to be addressed.

However the initial inspection of the visualisations revealed something unexpected. When the images of what is there now is compared with the digital reconstruction the building, at street level, is almost identical. What this shows, then, is that the building was actually completed to first floor level before construction stopped, and not to foundation level as originally thought. In its own right this proved to be a very enlightening piece of digital archaeology.

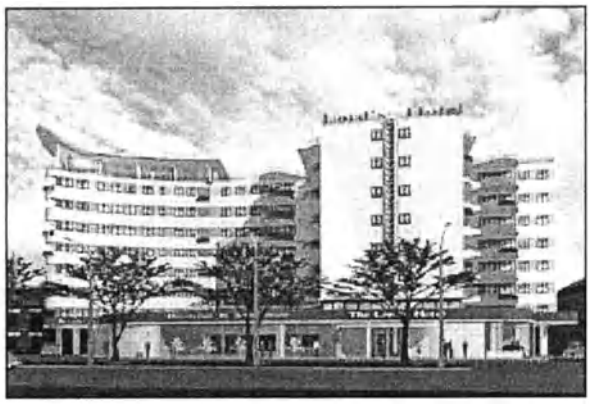


Figure 4. A faked contemporary photograph

Then there are other matters. What colour should the building be? A sepia photograph can be created (faked) from the digital data to show how it might have been recorded (Figure 4) but Connell Ward and Lucas used very strident colours in their other buildings internally and externally: sugar pink with royal blue, chocolate brown and lemon yellow. Unlike contemporary buildings this one would have challenged the convention of restrained use of colour. We can postulate what reaction would such a prominent building in such a particular location adjacent to Lords Cricket ground have drawn from architectural critics? Pink buildings were simply not cricket.

3.3 The postulated history and parallel debate

The different digital representations can now be used to develop an imagined architectural lineage and associated critical debate. As part of the scenario being constructed a report on the new Lords Court building in an issue of the Architects Journal in 1929 has been faked. A page of that imagined journal report is shown in Figure 5. This gives the scenario-building critic the opportunity to review and comment on the building that might have been, but with that discussion set in a contemporary context.

The building is likely to have been very contentious from several points of view. The scenario is therefore extended by suggesting that a group of prominent architects who were interested in what they regarded as a more restrained and socially responsible type of housing for the people would have made their objections well known.

Figure 5. A fake review: Architects Journal of 1929

This line of thinking leads to a faked letter in the Architectural Review in 1930 (see Figure 6) from a group of architects opposed to what they strongly believe to be an inappropriately strident addition to London's building stock.

Figure 6. A fake letter of objection printed in the Architectural Review 1930

By developing the argument in this way a debate set in the political, economic, cultural and architectural context of the time has to be invented. This requires some understanding of the relevant issues and context, and as such is a good and salutary exercise to run through. It creates an interesting framework through which an architectural argument can be formed (Rashidi, 1998). The line of postulation continues in this way, and it is assumed that

by the 1950s the building, like many 20s and 30s buildings, is in need of refurbishment. In the scenario proposed it is suggested that the building has been refurbished as a hotel. A report of the refurbishment now appears in a faked issue of the Architectural Record; see Figure 7.

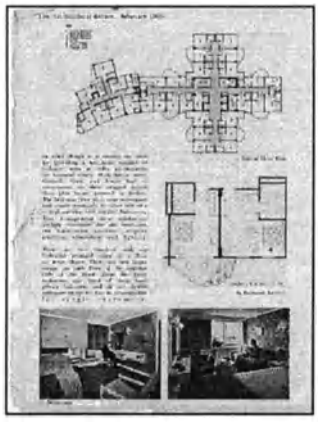


Figure 7. 1950s journal reporting on the refurbishment

The history of the building is imagined in a series of contemporary contexts in this way. In its most recently invented reworking the building becomes the base for a new cricket academy at Lords. As before the news of this event is faked (Figure8) and the report is created in the style of an issue of Building Design in 1998.

Figure 8. The building reborn as a cricket academy in 1998

From the 1920's, through to the end of the twentieth century a series of scenarios have been constructed, and, as shown here, these scenarios have been run out and made explicit through a series of faked journal articles.

The earlier stages of the process where the late 1920s through to the early 30s are covered show how the technique can be applied in a research setting. The digital archaeology showed that more of the original building had been constructed than had been first thought. But the modelling and visualisation of Lords Court made it possible to gauge the importance and potential impact that this building would have had. Its contribution to the portfolio of work that Connell, Ward and Lucas compiled would have been significant.

Extending the time line in the history of Lords Court from the 1920s through journal articles means that assumptions have to be made. It is the debatable nature of such assumptions that lead certain historians to question such lines of historical research. But the point of this technique is not simply about historical research; there are two other important reasons for its use. The first, like faking in art, is to provoke reaction. The second is to provide a framework for investigations as part of an architectural education. The simple fact is that to create a believable scenario the students has to understand the historical and architectural context in which the fake is set. For instance, in the example above the current issues and the way that they were described at the dates chosen for the faked journals had to be established. This application in an architectural education setting has been extended and an example of how this has been done is set out in the next section.

4. EXAMPLE TWO: EDUCATIONAL SETTING

It is important for architectural students to be able to stand back from their work and be self-critical. This is all part of the process of becoming a better designer. In other words, "On the one hand, the artist is the imaginer and producer. But he is also the critic" (Shahn, 1957). As an aside, one might say that, as critics, we could take issue with this quote and suggest that it should read "he or she is also the critic", but the essence of the point remains valid, that self-evaluation and reflection on one's own creative exploits is as important as the creative exploit itself.

As part of an architectural education Collins (1968) notes that it is vitally important for students to be made aware that the:

> "dialogue between his teacher and himself is simply an exercise in one aspect of the process of design, which he must learn to perform in solitude once the academic training is at an end. For there is no difference between criticism and self-criticism except for the number of people involved."

What digital technologies and virtual environments allow us to do is to create a framework for a richer kind of self-criticism. The example presented next is meant to illustrate how digital scenario-building can be used to foster a more interesting environment in which the architectural student can undertake such self criticism. It is, in fact, a way of moving on from the written essay or report that are rather stale as devices for students to report on and evaluate (in some cases their own) architecture.

The application of the technique described here was undertaken with third year students of architecture. These students were required to review a major project that they had completed, but to do so through a faked journal article. Choice of the journal to be faked was left to the individual student. The article was to be in two parts; the first a general review of the project, but presented as though it had been written by either a known architectural critic, or someone famous in an appropriate area. The second part was to be a technical review of the project in which the important aspects of the technology were to be reviewed and accounted for. Again, this was to be presented in the manner that the chosen journal presented such reviews.

The example presented below shows extracts from the submission by Paul Swarbrick; it is typical of the kind of response that the student group made. In Paul's case he chose to fake a contemporary edition of the Architectural Review, and the scheme reviewed was his combined emergency service centre (housing Police, Fire and Ambulance services).

Figure 9. Faked Journal

The front cover is shown in Figure 9, and a page from the main report appears in Figure 10. The report allows the author to review the scheme though the eyes of the fake critic, Timothy Brittain-Catlin, who talks about the creation of a 'unified and dignified unit'. The report allows the student to create a carefully constructed argument as to how they have created a

'unique and genuinely provocative scheme' in a way that is rarely possible in the conventional architectural 'crit'. Conventional presentations are spoken and because of this important points or lines of argument get lost. Even when presented as a written report the important issues can lose their impact. The fake review forces the writer to take a new standpoint; to imagine the response by an informed third party.

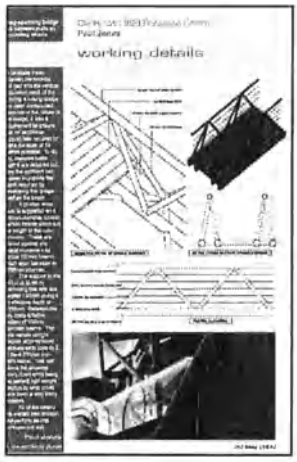


Figure 10. Typical page of the faked review

In terms of technical issues the condensed and focussed approach adopted by most journals forces the student to decide what is important, and what is critical in terms of the technology. Journals do not report on the routine, so the student has to decide what aspect(s) of the technology make their buildings special. A fake technical report, by another student, Paul Jones, is shown on Figure 10.

The results of the student exercise were very pleasing. The students felt that the faked scenario allowed them to make points and justify decisions in way that otherwise would not have been possible. In terms of the product, these were more engaging and pointed than those arising from more conventional ways of presenting a case.

5. CONCLUDING COMMENT

The idea of the fake as part of digital scenario building is developed in this paper. The aim was to show that, as a technique, scenario building presents an interesting and engaging way of developing an architectural critique. The emphasis of the work described here is to go beyond the interesting recreation of the visual aspects of the architecture and to

investigate the parallel influences on architectural criticism, and to take a third party view of an architectural issue. It is possible, then, for students and theoreticians to exploit virtual environments technology, using them to foster creative debate generated by deliberately and thoughtfully faked virtual histories.

It may occur to the reader that given all that has been said in this paper, a possible scenario is that the research work on Connell, Ward and Lucas never took place and the student project was never set. Maybe this paper is part of a faked scenario. Maybe.

6. REFERENCES

Attoe, W., 1978, *Architecture and Critical Imagination*, John Wiley & Sons, Great Britain.

Brown, A.G.P. and A.M. Simpson, 1997, "Historical Critique and Design Interrogation through CAAD", in *CAAD: Towards new design conventions*, Bialystok, August

Brown A.G.P. and Nahab, M., 1996, "Human interpretation of computer generated architectural images" *CAD and Creativity*, Bialystok, Poland, May.

Capaldi, N.1979, *The art of deception*, Prometheus Books, Buffalo (N.Y.)

Campbell, D.A 1995, "Vers Une Architecture Virtuelle", <http://www.hitl.washington.edu/people/dace/portfoli/arch560> October 1995.

Collins, P., 1968, " Philosophy of Architectural Criticism", Journal of the American Institute of Architects, 59, Jan. 1968, p. 49

Heeley, E., 1995, "Connell, Ward and Lucas: towards a complex critique", *PhD Thesis*, The University of Liverpool.

Hyett, P., 2000, The Architects Journal, June, UK.

Jones, M. 1992, *Why fakes matter: essays on problems of authenticity*, p.9, British Museum Press, London

Mitchell, W. 1992, *The reconfigured eye: visual truth in the post-photographic era*, The MIT Press.

Mitchell, W. 1999, Foreword to *Rendering real and imagined buildings : by Novitski, B.J.*, Rockport Publishers, Seattle

Novitski, B.J., 1998, *Digital Unbuilt*, Rockport Publisher, Seattle

Novitski, B.J., 1999, *Rendering real and imagined buildings : the art of computer modeling from the Palace of Kublai Khan to Le Corbusier's villas*, Rockport Publishers, Seattle.

Rashidi, H., 1998, *BArch dissertation*, School of Architecture and Building, University of Liverpool

Shahn, B., 1957, *The shape of content,* Harvard, Cambridge, p.34

Thistlewood, D and Heeley, E. 2001 (in press) *Connell, Ward and Lucas*, E & FN Spon, London.

Route analysis in complex buildings

Alexander Koutamanis[1 2], Marc van Leusen[1] and Vicky Mitossi[1]
Government Buildings Agency, Ministry of Housing, Spatial Planning and the Environment, The Hague[1], Faculty of Architecture, Delft University of Technology[2]

Key words: pedestrian circulation, analysis, representation, interaction

Abstract: Analysis of pedestrian circulation in buildings is usually performed in the early stages of the design process or later on with respect to a specific design problem such as fire safety. In both cases, the analysis relates more to wayfinding, i.e. search for a route on the basis of fundamental normative criteria. Wayfinding analysis in existing buildings is useful for the comparison between "rational" behaviour and actual usage but this comparison does little to explicate the observed structure of pedestrian circulation. In contrast to wayfinding, route analysis deals with the registration and assessment of actual patterns of pedestrian circulation in existing buildings. These patterns are represented topologically and geometrically. The geometric representation makes use of norms underlying building codes in order to reach an appropriate level of abstraction. Route representations are implemented on top of a building representation of relevant spatial and building elements. The building representation serves both as input and output for the route analysis. Input and output are largely automated, including production of the geometric route locally (i.e. within each space) and measurement of route distance and complexity. Use data are collected in an alphanumeric database and linked dynamically to the geometric and topological representation. Route analysis supports and refines other forms of post-occupancy evaluation by adding important dynamic aspects to activity allocation and compartmentalization.

1. PEDESTRIAN CIRCULATION IN BUILDINGS

It has been suggested that as a technological field matures its development gradually shifts from *illustrations* of fundamental ideas to *demonstrations* of how problems of practical significance can be solved and then to *experiments* that attempt to solve problems of practical interest

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 711–724.
© 2001 *Kluwer Academic Publishers.*

then to *experiments* that attempt to solve problems of practical interest (Winston, 1992). Despite the extent of the ongoing digitization of social processes and professional services, the computerization of architecture – especially in practice- still deals primarily with illustrations. One of the main problems in current developments is the limited integration, utilization and enrichment of domain knowledge. Human interaction with the built environment has been one of the knowledge components that are frequently underplayed in research, development and practice. The reasons for doing so include the domination of normative approaches to design and analysis (Koutamanis and Mitossi, 1996; Koutamanis, 2000), the complexity of human interaction, the aesthetization of architectural design and the lack of knowledge transfer in particular from cognitive science. In particular, dynamic aspects of the use of the built environment remain largely uncharted despite the ready availability of basic information on the cognitive, ergonomic and other dimensions of such aspects. One of these aspects is pedestrian circulation in buildings, which ironically is a primary even though frequently implicit factor in a variety of applications, such as the clustering and allocation of activities in a building (as in space allocation and programmatic analysis), code compliance (e.g. with respect to fire safety) and cost estimation.

Reasons for the relative neglect of pedestrian circulation in architectural computerization include:

- *The complexity of dynamic human interaction with buildings*: in moving from one space to another we process a wide spectrum of information, taking a large number of decisions and performing conscious and unconscious actions relating not only to the purpose of the movement but also to different contexts at the same time.
- *The complexity of computer simulations:* simulations for reproducing dynamic human interaction with buildings require high precision and accuracy, as well as computational power for implementation, testing and use.
- *Lack of comprehensive, integrated data* on dynamic human interaction with buildings: despite the availability of technologies for capturing human movement and the extent of relevant cognitive and ergonomic research, we possess no unified, verifiable body of data that could drive computer simulations of human movement in the built environment.
- *The relative weakness of programmatic and functional analysis:* the building brief is normally insufficient for design guidance with respect to pedestrian circulation or for the analysis and evaluation of designs produced on the basis of the brief. This is accentuated further by the normative abstraction of building codes and general professional

guidelines. Most analyses of designs and programmes appear to accept the reductive logic of such codes and guidelines.

- *Lack of integration with design synthesis:* the representations and other data used in analysis are frequently kept separate from the instruments and environments used in synthesis. As a result, providing input to the analysis becomes a tedious, redundant task. Even more important is the lack of direct feedback that constrains the further development of a design. Therefore, the designer currently views computational design analysis as a cumbersome, time-consuming alternative to intuitive analysis.

Our work focuses on the last two reasons. In the framework of the large, complex institutional buildings designed and managed by the Government Buildings Agency, conventional practices are frequently dismissed by all involved parties on the grounds that they are grossly insufficient for an accurate, precise and reliable projection of building behaviour and performance, without which it is impossible to take reasonable, defendable decisions. The extensive experience of clients, users and designers with the specific problems of governmental and institutional buildings makes the oversimplifications of intuitive and normative reasoning too weak to accept as fundamental common points of departure. At the same time, this experience provides a comprehensive (even though rather incoherent) empirical basis for the development of practical methods and instruments that make designing and communication efficient and transparent.

The simulation and analysis of a dynamic aspect such as pedestrian circulation relies on a representation consisting of a number of interrelated components:

1. *Actor(s):* person(s) who travel(s)
2. *Starting point:* the location from where the actor(s) depart. In buildings this location can be a space, a typical point in the departure space (such as its centroid) or a doorway. Multiple starting points indicate aggregation of routes / travels.
3. *Destination*: the location where the actor(s) go. Multiple destinations are not necessarily product of aggregation: a route / travel may also contain intermediate destinations. Also attractors such as stairs, lifts and control points can be treated as intermediate destinations.
4. *Path*: starting point, destination and locations traversed between the two. The path can be either the actual path of the route or an approximation of it (straight line or city block).
5. *Means of transportation*: how locomotion is performed along the path, including speed and capacity achieved by these means.
6. *Activities* that take place along the path:
    a. Activities related to transportation

b. Intervening opportunities, such as relations to other routes or other activities and actors in the building

The complexity and extent of the information in such a representation, as well as the corresponding tenacity of the analysis or simulation, mean that analysis of pedestrian circulation in buildings is usually performed either in the early stages of the design process when information and demands are quite abstract or later on with respect to some concrete design problem such as fire safety (Ozel, 1992). In both cases, the analysis relates more to wayfinding, i.e. search for a route on the basis of fundamental criteria such as the norms of building rules and regulations and basic contextual information. The ways the building is experienced, explored, understood and remembered by its users and occupants remains largely unexplored, even though it is an important factor in e.g. egress behaviour (Marchant, 1980; Shields and Silcock, 1987; Stollard and Abrahams, 1995). The reasons for the relative neglect of post-occupancy pedestrian circulation in research and practice are quite pragmatic. In addition to the already mentioned problems of complexity and lack of basic data, architecture deals with constraints that tend to be too abstract, as in code compliance. Detailed registration and analysis is therefore deemed useful only for the derivation of general principles and guidelines. This is accentuated by the lack of specificity in the form of a design and in the activities of its users in most design stages.

Wayfinding analysis in existing buildings is useful for the comparison between "rational" (normative) behaviour and actual usage, as well as for the refinement of predictive models of wayfinding for architectural design (Gross and Zimring, 1992; Shih and Yan, 1997). However, this comparison is generally transparent at the level of general principles or in very specific situations but as yet can do little to explicate the observed overall structure of pedestrian circulation, especially in large buildings. Recent extensions of basic wayfinding with techniques characterized by autonomy promise a correlation between local conditions and global structure. At low dimensionality (i.e. mostly concerned with internal interactions between pedestrians and flows) and at relatively high abstraction (urban design and higher) they provide useful insights into the potential of autonomous mechanisms in circulation simulation (Blue and Adler, 1998; Dijkstra, Timmermans et al., 2000). However, the complexity of interaction in real buildings appears to be too high for effective and reliable solutions yet.

Arguably the strongest point and at the same time the most important limitation of recent analyses and simulations of wayfinding behaviour lies in their attention for local decisions and actions. This is apparently a reaction to the determinism and reductionism of normative approaches that promises to enrich our understanding of cognitive factors in functional aspects and can lead to technical improvement in simulation and analysis. At the same time,

attention for local decisions and conditions may obscure global characteristics of a route, relations between routes and the conscious adaptation of routes for extrinsic reasons, such as in the arrangement of a travelling exposition in different exposition spaces or arbitrary compartmentalization of a building for security reasons. The current tendency of considering global dynamic patterns as mere products of local ones does little credit to both human adaptability and design capacity.

2. AN APPROACH TO ROUTE ANALYSIS

Our work follows a pragmatic bottom-up approach, intended to build on and enrich gradually existing professional practices. We employ a standard commercial CAD program (AutoCAD) for inputting and manipulating design representations that come close to conventional CAD drawings (Mitossi and Koutamanis, 1998). The use of these representations in design and analysis is structured and facilitated by a number of add-on modules (in Lisp and Visual Basic) for extracting information from the drawing representation, for performing analyses automatically and for checking grammatical or syntactic consistency and completeness (Koutamanis and Mitossi, 2000). The basic building representation consists of:

- *Spaces*: implemented in 2½D as polygonal outlines with a height indication (lightweight polylines with thickness)
- *Building elements:* also implemented in 2½D similarly to the spaces
- *A layer system* for the modularization of design information: clustering of spaces and building element by floor level and type
- *Alphanumeric annotations* on the spaces and building elements: these indicate properties that cannot be indicated graphically and are implemented as dynamic links to databases (in MS Access)

Relations between elements of the representation are recognized on an if-needed basis: when a control action or an analysis require information on e.g. relationships of inclusion or adjacency, the representation activates local detection routines that test specific relevant conditions. Most detection results are registered only temporarily in the framework of feedback mechanisms but often trigger other actions, such as a correction of the geometry of a space or building element. Automatic detection of relations and properties of spaces and building elements form the basis of most analyses. These are generally exported to prepared databases and spreadsheets (in MS Excel) where the actual analysis takes place (Leusen and Mitossi, 1998). The dynamic information system that comprises the representations, analyses and related background information, such as the building programme and cost data, is implemented as a Web-based

compound structure complete with communication facilities for use in the Intranet of the Government Buildings Agency or project-based Extranets.

In the framework of earlier research projects the analysis of pedestrian circulation has been considered with respect to the general structure of the design representation, as well as for specific functional issues (Koutamanis and Mitossi, 1993; Koutamanis, 1995). In accordance with our descriptive approach to design and analysis, these projects attempted to complement and question normative levels with a more detailed and accurate registration or projection of human interaction with the built environment.

In contrast to the predictions made in wayfinding analysis, route analysis deals with the registration and assessment of actual patterns of pedestrian circulation in existing buildings. The complex situations we experience in large, sensitive buildings such as prisons and law courts are so intricate that the normative levels are too coarse and deterministic to provide a comprehensive yet manageable picture that can be processed further by designers, occupants and decision takers. Moreover, the high specificity of such building types makes an extensive empirical analysis of existing buildings an obvious first step towards the specification of wayfinding and other simulations for use in the early design stages.

The initial application of route analysis in this framework formed part of a wider analysis of ten Dutch prisons in 1999-2000. It focused on basic routes common to all prison buildings that have substantial contribution to their management and operating costs, especially with respect to security facilities and personnel. This also makes such routes an important implicit factor in prison programming (e.g. for clustering composition and sizes), as well as for prison typology, which is based on assumptions underlying the (standardized) prison brief. Pedestrian circulation is furthermore important for user satisfaction. While the preferred length of route is a matter of prison regime and approach to supervision, complexity and lack of overview and control in the movement of prisoners is always undesirable.

The building representation of the prisons serves as basis for the topological registration of the routes as sequences of spaces linked by doors and other access-permitting openings. The topological representation is complemented by a geometric one that allows for a higher accuracy in the description of the route path. Input and output relations between route and building representations are largely automated, including production of the geometric route locally (i.e. within each space) and measurement of route distance and complexity. Use data (purpose of route, user and temporal information) are collected in alphanumeric databases and linked dynamically to the route representation. Also the analyses of routes individually and comparative analyses of different routes and different buildings in spreadsheets are linked dynamically to the spatial representations.

3. DESCRIBING A ROUTE TOPOLOGICALLY

A pedestrian route between two spaces *A* and *B* can be described topologically by the sequence of spatial elements that must be traversed so as to travel from *A* to *B*. This sequence comprises spaces and doors or similar openings permitting access. Consequently, a route can be represented by a subset of a building representation at two correlated levels, the topological and the geometric (Tabor, 1976). Topological route representations are closely associated with wayfinding. The basis of topological wayfinding is the access graph of a building, which is transformed into search tree by tracing all possible paths to the point they re-enter, previously visited nodes. Topological search for a path comes in two variations:

- *Satisfactory paths:* finding paths from starting positions to destinations by means of simple, basic procedures such as depth-first or breadth-first search. The length of the paths is generally not important.
- *Optimal paths:* finding shortest paths with procedures such as branch and bound, discrete dynamic programming or A*. In these the cost of traversing a path is of primary importance.

An interesting extension of the optimal paths variation is wayfinding with adversaries, as in e.g. games, by means of procedures such as minimax (Shannon, 1950) and alpha-beta pruning (Knuth and Moore, 1975). These provide possibilities for relating pedestrian movement with its context (intervening opportunities).

Also in our system the topological representation of a route is the starting point of registration and analysis. Rather than calculating the path on the basis of starting point and destination, the route data were registered in visits to the buildings and in interviews with the route users and controllers. The actual routes exhibit several marked differences with normative paths. The most important is that actual routes may contain loops. This is consistent with the security and supervision constraints of a prison. Loops in routing also exist in other building types such as exhibition spaces and should not be approached as a mismatch between circulation requirements and spatial arrangement. On the contrary, they can be features of specific activities. Another difference between the actual routes and some wayfinding and clustering approaches is that a route and its path are oriented: pedestrian circulation from *A* to *B* is not equivalent to the circulation from *B* to *A*. The removal of orientation that makes many formulations of a brief or of pedestrian circulation hopelessly static is not applicable to route analysis.

The implementation of the topological route representation posed a practical problem. While it derives from the tradition of graph-based representations of built space, the use of graphs on top of the building

representation would reduce the legibility of both the spatial and the route representation. Moreover, many users of the information system would have difficulty with graph representations. For these reasons, the topological representation was implemented as an alphanumeric annotation of a set of spaces (link to external database) and expressed visually as graphic annotations of the spaces by means of hatching. The sequence by which spaces are traversed in the route was not indicated in the annotation but, similarly to other relations in the representation, could be detected on an if-needed basis by the existence of direct access between spaces in the set of a specific route. The detection of loops is not included in this version of wayfinding, as it requires closer correlation with extrinsic, yet uncharted constraints such as visual and electronic supervision.

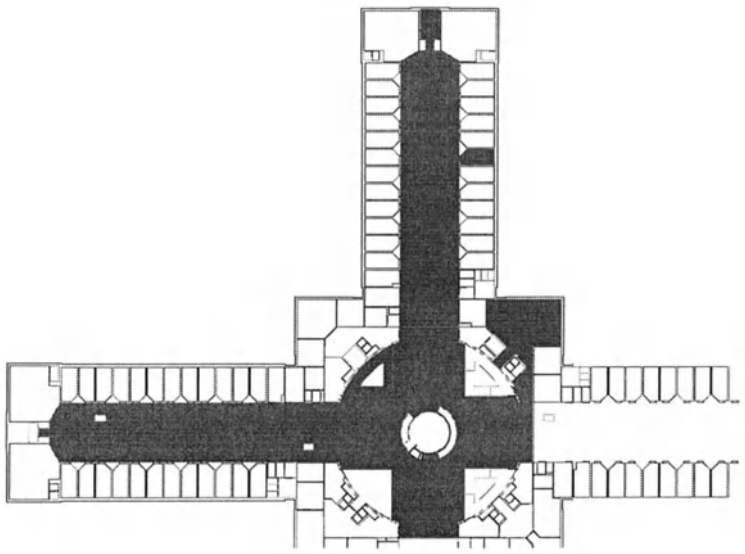


Figure 1. Topological route representation

4. DESCRIBING A ROUTE GEOMETRICALLY

Any elaboration of the topological representation of a route (with the exception of some temporal aspects) involves transition to a geometric representation where the path of the route is described with accuracy and precision. The geometric representation is obviously more significant within the nodes of a topological representation, i.e. within the spaces of the route. The size of most doors and similar openings is such that any perturbation of the path as it passes though them is insignificant for the overall route. This does not preclude local ergonomic or cognitive problems, e.g. in opening a

door. However, such problems depend more on detailing and are generally as invisible in the geometric representation as in the topological one. Alphanumeric or other annotations are probably the best means for registering these problems on the route representations.

The geometric path in a space has been a traditional hard problem in both wayfinding and route analysis. The usual focus has been obstacle avoidance. Nevertheless, we should not forget the influence of intervening opportunities of e.g. interaction with the building and its occupants, as well as relations with other routes that may cause congestion or disorientation. As a result, models of spatial reasoning, i.e. reasoning about motion in space have to move beyond avoiding obstacles and moving around obstacles (Brady, Hollerback et al., 1982; Brooks, 1983; Lozano-Pérez, 1983) to be really applicable in the analysis of buildings and designs. Recent research into the applicability of autonomous mechanisms and interactive immersive environments for wayfinding seems promising in this respect (Shih and Yan, 1997; Dijkstra, Timmermans et al., 2000).

In a conscious attempt to reduce the complexity of their subject matter, conventional analyses of pedestrian circulation have employed approximations of the path by means of techniques such as the straight line or the city block. In our analyses ascertaining the actual path is also too time-consuming and uncertain to be a viable option. Even under the carefully controlled conditions of a prison, human choice and contextual variability conspire to create a spectrum of local alternatives and variations. For such reasons we have opted for a normative basis derived from the definition of the shortest path in current Dutch building regulations, which has been previously applied to fire escape analysis (Koutamanis, 1995). The essential characteristic of this definition with respect to obstacle avoidance is that the shortest path should have a distance of at least thirty centimetres from fixed obstacles. Also the starting point of a path should be the furthest point from the exit in the departure space, again thirty centimetres away from fixed obstacles such as walls.

Our basic adaptation of the normative definition attempts to neutralize the slight bias towards the destination side that results from the thirty-centimetre constraint. The starting point is assigned to the centroid of the departure space and the path passes openings in the middle. All other obstacles were kept at a distance of thirty centimetres. The user is allowed to add to the regulatory fixed obstacles, so as to improve the precision of the average route path. The resulting geometric representation elaborates locally the connection of two links in the topological representation through the connected node: the local path starts at the centroid of the first link (opening) and ends at the centroid of the other link, passing though the node (space)

and keeping within this space a distance of thirty centimetres from all objects that should be treated as obstacles.

The explicit representation of relevant entities in the building representation and the use of local intelligence permits a large degree of automation in the derivation of the route paths. User input is limited to the selection of the two openings that should be linked by a local path and this only if a space has more than two openings. The only other possibility for user interaction concerns obstacle avoidance: if a path comes closer than thirty centimetres to an obstacle, the route analysis system does not attempt a shortest path calculation around the obstacle but refers the problem to the user. With the initial path and its intersections with a thirty-centimetre no-go area around the obstacle as basis, the user traces over the local path that agrees most with the actual route.

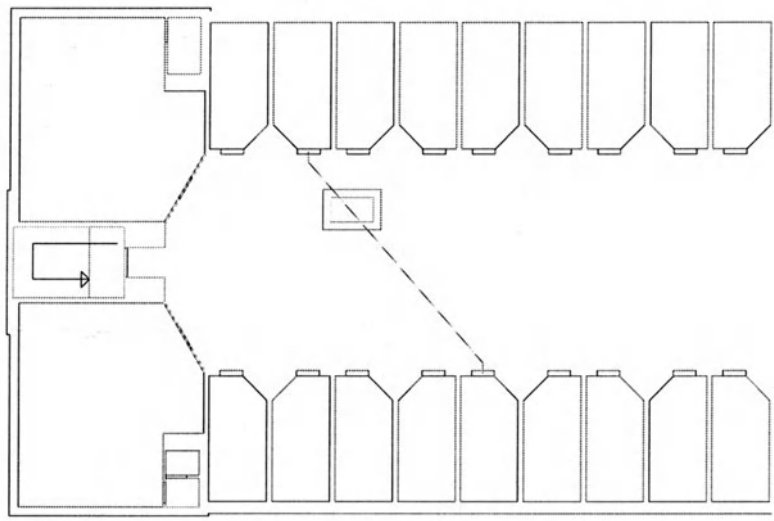


Figure 2. Local path and obstacle in the geometric route representation

Also the integration of local paths into the complete geometric representation of a route is based on local recognition. As local paths connect with each other at the centroids of openings, the topological representation is used to collate them together into the geometric representation of the route.

Non-horizontal paths are often treated as a weighted version of their horizontal projection. However, a prerequisite to the use of coefficients for the adaptation of distance, time and energy spent along a non-horizontal path of the route is justification of the ergonomic and cognitive adaptation factors and validation of the results. The lack of such data meant that we opted for a

geometrically accurate three-dimensional representation of the path that allowed for the identification of vertical as well as horizontal displacement along the route.

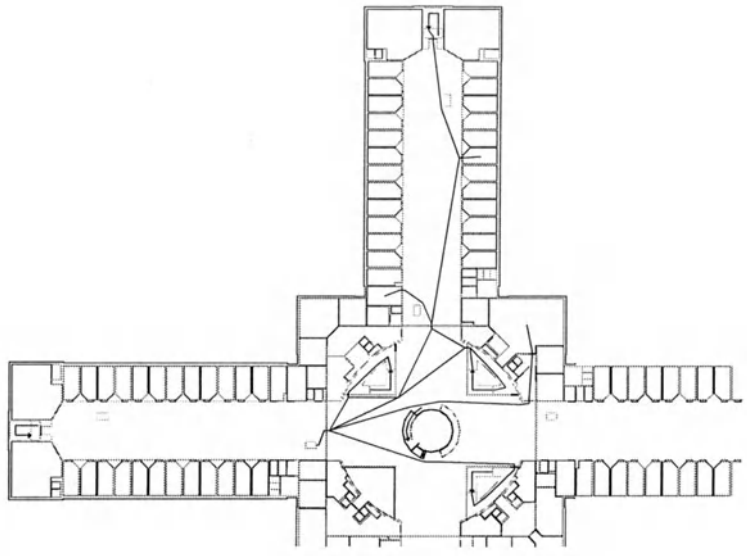


Figure 3. Geometric route descriptions

5. FURTHER DEVELOPMENT

The route analysis system described in the present paper is primarily meant as an information-management component of a wider design information system. In its current form it provides a reasonably accurate and comprehensive description of pedestrian circulation in quite complex buildings. Moreover, this description is at an appropriate level of abstraction for serving as communication matter in the discussions between designers, decision takers and building users. In the framework of the prison analysis of 1999-2000 it was used effectively to assess pedestrian circulation in each prison individually and to compare similar routes in different buildings. The ability to analyse and compare with respect to both the topology and the geometry of routes, allowed for different measures of complexity and distance. Moreover, the combination of qualitative and quantitative data facilitated the exploration and presentation of the effects of wider programming and typologic constraints on prison design in The Netherlands. Such effects are not a matter of a global measure of circulation cost but of precise identification and substantiation of observed and potential problems, both within a route and in the relations between separate routes. In existing

buildings an accurate description of such problems provides a responsive analysis of ideas concerning the improvement of facilities management, activity allocation, operational cost reduction and redevelopment of a building. The same information can also influence the design of similar new buildings as a source of examples for brief improvement and testing, as well as of cases for refining design thinking.

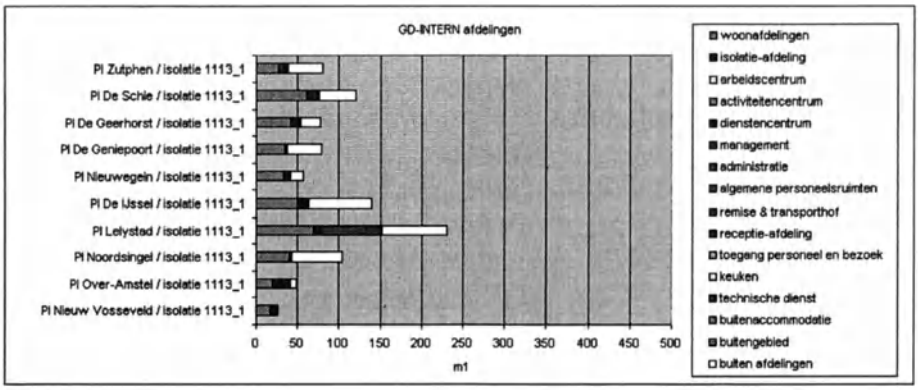


Figure 4. Route comparison

From a methodical and technical viewpoint, the route analysis system provided useful insights into the possibilities for adapting wayfinding research results to the prediction of realistic routes. For example, the effects of compartmentalization, security barriers and heuristics for e.g. avoiding spatio-temporal coincidence of distinct routes are such that automatic shortest path calculation either becomes too complex and therefore too unreliable without user intervention. A promising direction for further development is he addition of memory components that represent knowledge of the specific building (Gross and Zimring, 1992), as well as of the particular rules and constraints of prison supervision and security.

Such memory components relate closely to local intelligence and in particular to autonomous mechanisms that reproduce or explore wayfinding behaviour in the building representation. Prior experience with such mechanisms suggests that they are helpful for the representation of temporal aspects and for the exploration of capacity issues (Koutamanis, 1995) but require a more global knowledge of the building and the route in order to go beyond local refinement. For example, finding an optimal path in a route with intervening opportunities and conflicts presupposes reliable avoidance of the horizon effect, i.e. recognition of lesser, temporary sacrifices in the context of wider concerns and gains (Berliner, 1978; 1980).

On a practical level, further development of the route analysis system should address its two major limitations for information processing:

1. *The temporal dimension* (including speed of movement), which is currently absent from the system not in terms of registrable information but of simulation possibilities that facilitate comprehension and communication of pedestrian circulation patterns.
2. *Representation of capacity and volume* in the routes and the sequence of spaces and openings that accommodate them. These are arguably the most important omissions in the current information system, as they provide essential indications of functional interaction with building and related performance measures.

6. REFERENCES

Berliner, H.J. (1978). "A chronology of computer chess and its literature." *Artificial Intelligence* 10(2).

Berliner, H.J. (1980). "Computer backgammon." *Scientific American* (June).

Blue, V.J. and J.L. Adler (1998). "Emergent fundamental pedestrian flows from cellular automata microsimulation." *Transport Research Board* (1644): 29-36.

Brady, M.J., J.M. Hollerback, et al., Eds. (1982). *Robot motion: planning and control.* Cambridge, Massachusetts, MIT Press.

Brooks, R.A. (1983). "Planning collision free motions for pick and place operations." *International Jurnal of Robotics Research* 2(4).

Dijkstra, J., H.J.P. Timmermans, et al. (2000). Towards a multi-agent system for visualizing simulated behaviour withing the built environment. *Design and decision support systems in urban planning. Proceedings of the 5th International Conference.* Eindhoven, Eindhoven University of Technology.

Gross, M.D. and C. Zimring (1992). Predicting wayfinding behavior in buildings: a schema-based approach. *Evaluating and predicting design performance.* Y. E. Kalay (ed.). New York, Wiley.

Knuth, D.E. and R.W. Moore (1975). "An analysis of alpha-beta pruning." *Artificial Intelligence* 6(4).

Koutamanis, A. (1995). Multilevel analysis of fire escape routes in a virtual environment. *The global design studio.* M. Tan and R. The (eds.). Singapore, Centre for Advanced Studies in Architecture, National University of Singapore.

Koutamanis, A. (2000). "Digital architectural visualization." *Automation in Construction* 9(4): 347-360.

Koutamanis, A. and V. Mitossi (1993). On the representation of dynamic aspects of architectural design in machine environment. *Advanced technologies. Architecture, planning, civil engineering.* M. R. Beheshti and K. Zreik (eds.). Amsterdam, Elsevier.

Koutamanis, A. and V. Mitossi (1996). Simulation for analysis: Requirements from architectural design. *Full-scale modeling in the age of virtual reality.* B. Martens (ed.). Vienna, OKK.

Koutamanis, A. and V. Mitossi (2000). Grammatical and syntactic properties of CAAD representations for the early design stages. *Design and decision support systems in architecture. Proceedings of the 5th International Conference.* Eindhoven, Eindhoven University of Technology.

Leusen, M.v. and V. Mitossi (1998). A practical experiment in representation and analysis of buildings. *4th Design and Decision Support Systems in Architecture and Urban Planning Conference.* Eindhoven.

Lozano-Pérez, T. (1983). "Spatial planning: A configuration-space approach." *IEEE Transactions on Computers* 71(7).

Marchant, E.W. (1980). Modelling fire safety and risk. *Fires and human behaviour.* D. Canter. Chichester, Wiley.

Mitossi, V. and A. Koutamanis (1998). Spatial representations as the basis of formal and functional analysis. *4th Design and Decision Support Systems in Architecture and Urban Planning Conference.* Eindhoven.

Ozel, F. (1992). An intelligent simulation approach in simulating dynamic processes in architectural environments. *CAAD Futures '91 —Computer Aided Architectural Design Futures. Education, research, applications.* G.N. Schmitt. Wiesbaden, Vieweg.

Shannon, C.E. (1950). "Automatic chess player." *Scientific American* 182(48).

Shields, T.J. and G.W.H. Silcock (1987). *Buildings and fire.* Harlow, Essex, Longman.

Shih, N.-J. and C.-S. Yan (1997). A study of the location of fire egress signs by VR simulation. *CAAD Futures 1997.* R. Junge (ed.). Dordrecht, Kluwer.

Stollard, P. and J. Abrahams (1995). *Fire from first principles. A design guide to building fire safety.* London, E. & F. Spon.

Tabor, P. (1976). Analysing route patterns. *The architecture of form.* L. March (ed.). Cambridge, Cambridge University Press.

Winston, P.H. (1992). *Artificial Intelligence.* Reading, Massachusetts, Addison-Wesley.

Designing for Interest and Novelty

Motivating Design Agents

Rob Saunders and John S Gero
The University of Sydney

Key words: design agents, novelty, curiosity, learning

Abstract: This paper is concerned with the motivation of design agents to promote the exploration of design spaces. A general form of motivation common to designers is a curiosity to discover interesting designs. This paper presents computational models of interest and curiosity based on the detection of novelty. We illustrate the behaviour of our model of interest by developing a design agent that is motivated to explore the effects of emergent crowd behaviours on the performance of doorways.

1. INTRODUCTION

The search for interesting designs is a primary motivation for designers. Interesting designs provide information about the design task and allow the designer to learn in advance of a need to apply the knowledge. This type of curious self-directed learning plays an important role in the weaving together of problem finding and problem solving, within and between design sessions.

Studies of preference judgements in designers and non-designers show that the subjective determination of interestingness depends upon the previous experiences of the individual (Whitfield and Wiltshire, 1982; Purcell and Gero, 1992; Martindale, 1990). A design is most likely to be considered interesting if it is similar-yet-different to previously experienced designs. In other words, a design is likely to be interesting if it is novel but not entirely unfamiliar. Consequently, a motivation to seek out novelty can be a useful general-purpose heuristic in design. Martindale (1990) proposed that the search for novelty is the only constant motivation in the

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 725–738.
© 2001 *Kluwer Academic Publishers.*

development of artistic and architectural styles as cultural and social conditions change over time.

1.1 Emergence in Design

One source of novelty familiar to designers is emergence. A property of a design that is not represented explicitly at the time of creation is said to be an emergent property if it can be made explicit (Gero, 1994b; Mitchell, 1993). *Design emergence* is the process of recognition and explicit representation of emergent properties (Gero, 1994a). A familiar example of design emergence is shape emergence.

Emergent shapes in a drawing or sketch are unintended consequences of the drawing actions that produced them (Schön and Wiggins, 1992). Protocol studies of designers while sketching have shown that unexpected discoveries of emergent shapes can have a significant impact on the course of further design activity (Schön and Wiggins, 1992; Suwa, Gero et al. 1999). *Shape emergence* is the process of recognition and explicit representation of emergent shapes.

1.1.1 Computationally Modelling Shape Emergence

Typically, computational models of shape emergence have created an unstructured intermediate representation of a sketch and then identified emergent shapes by combining elements of the intermediate representation in new ways. Computational systems using infinite maximal lines (Gero and Yan, 1993) have proved successful in identifying emergent shapes (Damski and Gero, 1996) and emergent shape semantics (Gero and Jun, 1995). Alternative computational models of shape emergence have used bitmap images as intermediate representations and image processing techniques to find emergent shapes. Liu (1993) used neural networks to identify previously learned emergent sub-shapes, Tomlinson and Gero (1997) used a neural model of early visual processing, and Edmonds and Soufi (1992) used Gestalt operators to construct emergent groupings of shapes.

1.1.2 Computationally Modelling Design Emergence

Shape emergence is not the only form of design emergence that can be computationally modelled. Also, to exploit emergence in future design tasks, design agents must learn about the initially unintended consequences of their actions. Most of the computational models of shape emergence have lacked the ability to learn. As a consequence all of the emergent shapes discovered had to be considered potentially "interesting" and presented to a user for

further evaluation. In contrast, the computational model of design emergence presented here builds on previous work that developed a model of shape emergence capable of learning to expect emergent shapes (Gero and Saunders 2000).

The computational model of interest described below has been developed in recognition of the fact that design emergence is more than shape emergence: it models interest in the emergence of unexpected group behaviour in crowds of simulated pedestrians. The model of interest is used to develop a computational model of curiosity that uses the evaluation of interestingness to motivate the actions of a design agent. The task of the curious design agent, in this example, is to explore a space of possible doorway designs that allow crowds of simulated pedestrians to pass in opposite directions.

While the problem of designing a doorway is conceptually simple, the complex interactions between the pedestrians mean that emergent group behaviours play a critical role in determining the performance of different designs. Therefore the initial statement of the design problem is necessarily ill defined: it *cannot* include a description of every relevant detail of emergent group behaviour in advance. This provides a similar problem to those faced by human designers: our design agent's task includes both problem finding and problem solving.

Section 2 introduces our approach to developing curious design agents. Section 3 describes some experiments with an implementation of a curious design agent applied to the design of a doorway. We conclude with a discussion of the potential benefits of using curious design agents to assist human designers.

2. DEVELOPING CURIOUS DESIGN AGENTS

In this section we describe our approach to developing curious design agents. We begin by examining the role that curiosity and interest can play in computational models of designing. We then describe the components of a curious design agent.

2.1 Curiosity

In humans and animals the drive that we call curiosity rewards self-directed learning through inquisitive exploration in advance of a need to apply the knowledge gained. Berlyne (1971) describes curiosity as follows:

> Uncertainty can generate a kind of motivational condition that we call "curiosity". [...] It will impel action to obtain further information from,

or relating to, the object of curiosity so that information capable of relieving the uncertainty can be absorbed.

Curiosity motivates a designer to explore interesting designs to relieve the uncertainty that accompanies an incomplete understanding of the design space. A designer can be motivated by curiosity to investigate a new approach to solving a problem simply because it is interesting rather than because it is successful. Alternatively, curiosity can motivate a designer to explore new problems because the designer can recognise interesting problems where familiar designs do not perform as expected, whether for the better or for the worse.

Computationally speaking, *curiosity is a process that internally generates reinforcement signals sent to an agent's controller that rewards the discovery of interesting concepts.* The main difference between curious agents and other types of reinforcement learning agents is that some of the reinforcement signal is generated internally to reward the discovery of novelty (Schmidhuber, 1991). Curious design agents must be able to recognise both problems and solutions as interesting, fortunately, the same mechanisms can be used for both types of recognition.

2.2 What's Interesting?

In general, determining interestingness depends upon the knowledge of the agent and their computational abilities; things are boring if either too much or too little is known about them (Schmidhuber, 1997). Hence situations that are similar-yet-different to previously experienced situations are the most interesting and this is what we mean when we say that something is novel. *A novel situation is one that is similar enough to previous experiences to be recognised as a member of a class but different enough from the other members of that class to require significant learning.*

It is a relatively straightforward to develop a computational model of interest based on this definition of novelty. A very simple model of interest used in the following experiments maintains an average of the novelty detected over a fixed window of the ten most-recent situations. A boredom threshold is used to determine when the interest in the current area of a design space is low enough to warrant a change in the design process, e.g. a switch from problem solving to problem finding.

Empirical research suggests a strong connection between novelty and aesthetic preference in various creative fields including literature, art, architecture and music (Martindale, 1990; Gaver and Mandler, 1987). These reports lend weight to the argument that novelty is an important determinant of interest in many creative fields including architecture.

2.3 A Curious Design Agent

The implementation of a curious design agent described here uses a combination of neural networks and reinforcement learning. Neural networks are used to construct a world model, i.e. a mapping from designs to evaluations. The world model constructed by the neural networks is rarely perfect and predictions of evaluations from design descriptions often contain errors. Some errors stem from a lack of adequate training and are of little interest but others are potentially more important.

Two sources of potentially interesting errors found in designing are consequences of emergent properties of the design task and the nature of learning and recall processes. Emergent properties of a design task can be sensitive to small differences in design parameters that can make a big difference to performance. World models that do not take these small differences into account can contain significant errors. Machine learning algorithms trade off being plastic enough to learn about new experiences with being stable enough to recall memories of previous experiences. The balance struck between stability and plasticity can have a significant affect on the accuracy of predictions.

A process called novelty detection is used to determine a measure of novelty that is proportional to the amount of error in the predictions of the neural networks. The level of interest in the current area of design space is calculated from the novelty that is used to produce a reinforcement signal for the agent's controller. The goal of a curious design agent is to maximise the reinforcement signal by seeking out novel situations.

2.3.1 Novelty Detection

The purpose of novelty detection is to identify unexpected or abnormal situations from examples of normal behaviour. Novelty detection has been used in domains as varied as medical diagnosis (Tarrasenko, 1995), industrial plant monitoring (Worden, 1997), robot navigation (Marsland et. al., 2000) and text retrieval (Yang, 1998).

Our implementation of novelty detection uses two Habituated Self-Organizing Maps (HSOMs) to estimate the novelty of a situation. An HSOM consists of a standard self-organising map (SOM) with an additional neuron that outputs the novelty of the current input (Marsland et al., 2000).

A self-organising map consists of a lattice of neurons that are used to represent different categories of inputs (Kohonen, 1993). Each neuron has an associated vector of weights of the same dimension as the inputs. When a new input is presented to the SOM each neuron compares the similarity of its weight vector to the inputs. The neuron with the best matching weights is

declared the winner. Learning is accomplished by updating the winner to reduce the difference between its weights and the inputs. In addition a neighbourhood of neurons around the winner are updated to reduce the difference between their weights and the inputs. This process results in a topographic map of the input space, with similar categories being represented by nearby neurons.

In an HSOM every neuron in the SOM is connected to the output neuron by habituating synapses that become less effective at transferring activation between neurons with use. The more frequently a map neuron fires the lower the efficacy of the synapse and hence the lower the output of the novelty detector.

The first HSOM estimates the novelty of a design by categorizing a representation of the design solution. The second HSOM estimates the novelty of the performance of the design by categorizing a profile of the design situation that includes representations of the design solution, the design problem and an evaluation of the design's performance.

The inverse of the novelty detected by the first HSOM is used to estimate the familiarity of a design. The output of the second HSOM is used to estimate the novelty of the design performance. The novelty of a design situation is calculated as a product of the familiarity assigned by the first network and the novelty assigned by the second. Consequently, significant novelty is only detected when a familiar design has an unfamiliar performance.

3. DESIGNING VIRTUAL ENVIRONMENTS FOR SIMULATED PEDESTRIANS

A simple crowd management problem is used to illustrate the behaviour of our curious design agent. The problem is to design a doorway to facilitate the efficient and comfortable movement of crowds of pedestrians travelling in opposite directions. A pedestrian simulator was developed to evaluate doorway designs. Pedestrian movement is simulated using a microscopic model of crowd behaviour developed to account for empirically observed self-organising phenomena.

3.1 Simulating Pedestrians

Computer models of pedestrian movement have been used to provide valuable tools for designers when planning or modifying pedestrian areas in large buildings like railway stations or shopping malls (Major et al., 1998). The "social force model" is a microscopic model of pedestrian behaviour

that simulates the behaviour of individual pedestrians to model self-organising phenomena in crowds (Helbing, 1991).

3.1.1 The Social Force Model

Helbing and Molnár (1995) developed the social force model of pedestrian behaviour to simulate the pedestrian crowd movements to gain a better understanding of empirical results. The "social forces" in the model do not represent forces exerted upon a pedestrian; rather they are an approximation of the internal motivations of the individuals to move in certain directions. Despite its simplicity, computer simulations have shown that the social force model is capable of realistically describing several interesting aspects of collective pedestrian behaviours observed in empirical studies (Helbing and Molnár, 1997). The social forces modelled in these experiments are listed in *Table 1*. Detailed mathematical descriptions of these forces can be found in Helbing and Molnár (1995).

Table 1. The social forces modelled in the simulations of pedestrian crowds.

	Description of social force
1.	Pedestrians are motivated to move as efficiently as possible to a destination.
2.	Pedestrians wish to maintain a comfortable distance from other pedestrians.
3.	Pedestrians wish to maintain a comfortable distance from obstacles like walls.
4.	Pedestrians may be attracted to other pedestrians (e.g. family) or objects (e.g. posters).

3.1.2 Evaluating Virtual Environments

Designs are evaluated using measures of the efficiency and discomfort for each simulated pedestrian (Helbing and Molnár, 1997). Efficiency is measured for a pedestrian as the average difference between actual walking speed during a simulation and desired walking speed. Discomfort is calculated as a function of the number of direction changes during a simulation that a pedestrian must perform in order to negotiate the built environment and other pedestrians.

Like an architect, the primary concern of our design agent is the "subjective experience" of the simulated pedestrians visiting its environment. However, it should be stressed that our curious design agent does not attempt to optimise its designs in the computational sense. Instead the design agent is motivated to explore the space of possible designs. It is equally motivated to investigate good and bad designs, e.g. inefficient designs can be interesting if their inefficiency is unexpected.

3.2 Experimental Results

This section describes two experiments using the models of interest and curiosity described above. The first experiment investigated the detection of novelty as emergent group behaviours affect the performance of three doorway designs. The second experiment investigated the behaviour of a curious design agent autonomously exploring the problem and solution spaces of doorway design.

3.2.1 Experiment 1: Assessing the Novelty of a Two Door Design

To illustrate the performance of the novelty detector, three designs for a doorway were created. The three doorway designs were for a narrow door, a wide door, and a combination of two narrow doors, as shown in *Figure 1.*

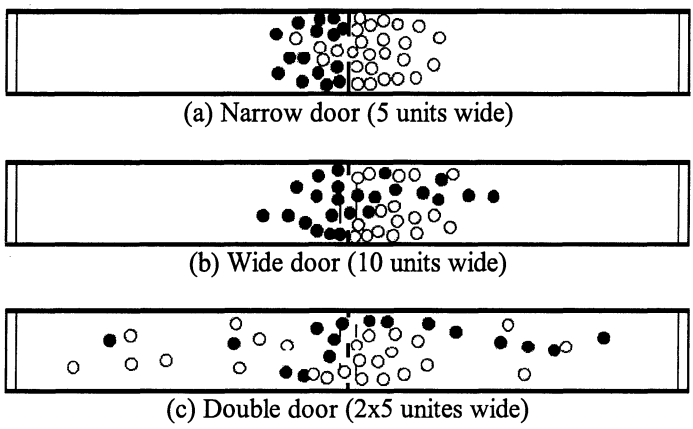


Figure 1. Screenshots of the simulations of pedestrian flow through (a) a narrow, (b) a wide, and (c) a double doorway design with a crowd of 40 pedestrians. The black circles indicate pedestrians travelling from left-to-right across the doorway and the white circles indicate pedestrians moving from right-to-left.

The doorway designs were tested using different numbers of pedestrians simultaneously trying to get through the doorway, crowds ranged in size from 1 to 51 pedestrians in increments of 10. The efficiency and discomfort measures from the simulations were combined into a single evaluation measure for each simulation. The best evaluations of three trials conducted at each crowd size are shown in *Figure 2.*

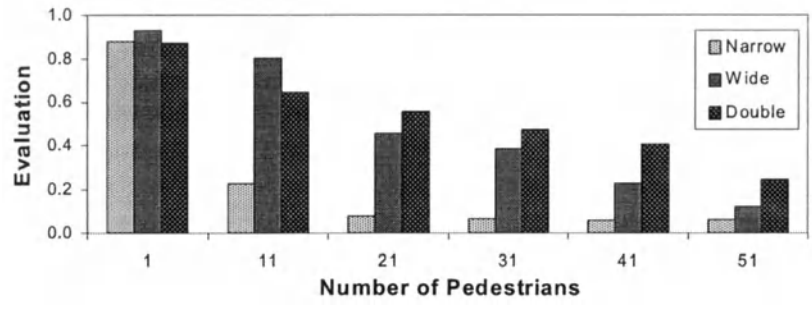


Figure 2. The best combined efficiency and discomfort evaluations for narrow, wide and double doorway designs for different crowd sizes (1–51 pedestrians).

All doorways performed equally well with only one pedestrian passing through it at a time. As the number of pedestrians increases the crowds display an oscillatory behaviour around doorways where one group of pedestrians gains control of the whole door at a time. The control of the doorway switches back-and-forth in direction as the numbers of pedestrians on either side of the doorway change.

The performance of the narrow doorway design quickly deteriorates to give consistently bad evaluations as the number of pedestrians increase. The wide doorway design maintains a very high performance for 11 pedestrians but its performance reduces dramatically, by almost 30%, as the number of pedestrians increases to 21. The performance of the wide door degrades more slowly over as the crowd sizes continue to increase from 31–51 pedestrians.

The performance of the double doorway design degrades even more slowly than the wide doorway design. For small crowds with less than 11 pedestrians the wide doorway design performs better but as the numbers of pedestrians increase the double doors outperform the wide door.

The double doorway design's superior performance in crowded conditions is a consequence of an emergent organisation. The two doors become specialised in the transfer of pedestrians moving in a single direction for relatively long periods of time. This can be seen in the double doorway simulation shown in *Figure 1*, pedestrians travelling from left to right pass through the top door while pedestrians travelling right to left pass through the bottom door.

The evaluations of each doorway design were presented to the novelty detector in ascending order of pedestrian numbers. The evaluations of the narrow doorway were presented first, the wide doorway evaluations second and the evaluations of the double doorway were presented last. The best novelty measures of three trials are presented in *Figure 3*.

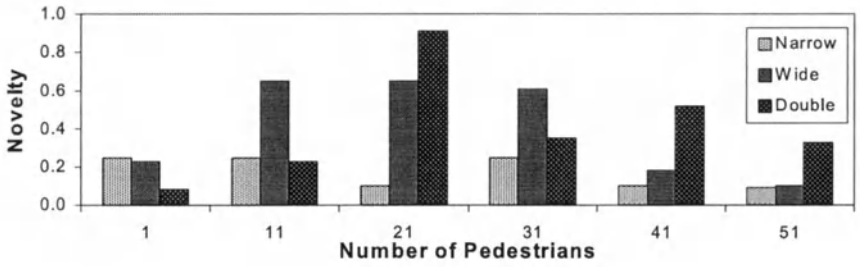


Figure 3. The greatest novelty detection for narrow, wide and double doorway designs for different crowd sizes (1–51 pedestrians).

Very little novelty was detected for the narrow doorway design at any crowd size. This is due to the lack of experiences against which the novelty detector could compare performances and the fact that the narrow doorway had consistently bad performance with more than one pedestrian.

The relatively high (~0.6) novelty measure for the wide doorway simulations with 11, 21 and 31 pedestrians indicate the improved performance of the wide doorway over the narrow doorway. The novelty of the wide doorway design drops at larger crowd sizes as the characteristics of the wide doorway are learned.

The novelty assessments of the double doorway design show very high novelty measures for simulations using 21 pedestrians, highlighting the resistance of the double doorway design to the fall in performance suffered by the wide door. The subsequent levels of novelty for simulations involving 31, 41 and 51 pedestrians reflect the relative differences in evaluations as the advantages of the double door design are maintained and the characteristics of the new design are learned.

The results of this experiment show that novelty detection can identify the most interesting designs without extensive reasoning by comparing the relative performance of different designs under similar conditions. The same novelty detector was used in the next experiment to implement models of interest and curiosity for an autonomous design agent.

3.2.2 Experiment 2: Curious Problem Finding and Problem Solving

In this experiment a curious design agent was given two conceptual spaces to explore: a problem space and a solution space. The solution space was defined by two variables: the number of doors making up the doorway and the combined width of doors. The problem space was defined by a single variable: the total number of pedestrians in the two crowds trying to

get through the doorway. All other variables of the simulation remained constant.

Figure 4 shows the novelty detected over the course of a design session. The design agent was initially given a narrow doorway as a solution to the problem of moving a single pedestrian. The novelty of exploring this design soon decreases as the agent learns to accurately predict the doorway's performance, the agent's interest level quickly falls below its boredom threshold and it begins to explore the problem and solution spaces for more interesting situations.

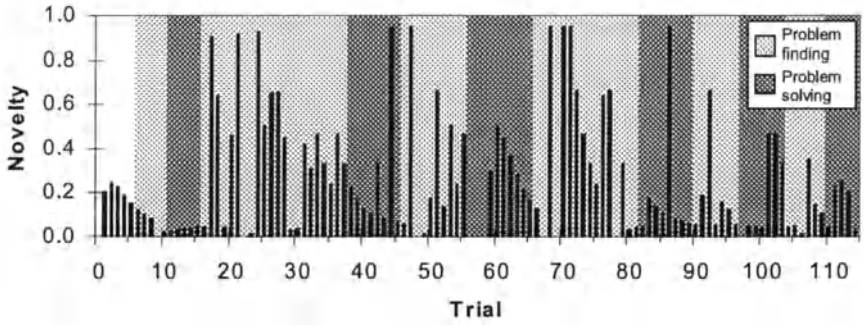


Figure 4. The results of using a curious design agent to explore the problem and solution spaces for doorway design. The chart shows the novelty detected for each simulation trial. The light shaded regions indicate that the design agent is problem finding and dark shaded regions indicate that the design agent is problem solving.

Figure 4 shows the design agent switching between searching the problem and solution spaces as interest in a particular problem or solution wanes. The chart shows the "tailing-off" of novelty values as the characteristics of situations are learned. It also shows how the detection of novelty extends the period that an agent spends searching a particular space, especially the exploration of the problem space for trials 17–37 and 67–82.

The highest peaks in detected novelty (~0.9) in the first half of the experiment (up to trial 68) all correspond to simulations using double doorway designs as these have significantly different characteristics to single doorway designs initially explored.

The high peaks in the second half of the chart correspond to simulations using wide doorway designs. This change in fixation occurs when the interest in double doorway designs subsides. In the second half of the design session the design agent is discovering an array of interesting situations where a wide door does not perform in the same way as a double door. At lower numbers of pedestrians the wide doorway does better than the double doorway, while at higher numbers of pedestrians it performs worse. Either

way, the design agent finds situations involving wide doorway designs novel and maintains a higher level of interest in exploring this area of the design space than would otherwise be expected.

The change in fixation of the design agent from double to wide doorways illustrates a difference in exploration between a more conventional optimisation approach and one based on curiosity. The curious design agent did not explore the situations using wide door designs because they performed better than the double door designs. Instead, it explored the space of wide door designs because they did not perform as expected from previous experiences of the similar-yet-different double door designs.

4. DISCUSSION

The experimental work described has investigated models of interest and curiosity using processes that detect the novelty of similar-yet-different design situations. Experiment 1 showed that novelty detection could be used to identify interesting situations where unexpected emergent properties play an important role in the evaluation of designs. Experiment 2 showed that using this model of interest a curious design agent can autonomously explore problem and solution spaces to identify interesting design situations.

Future experiments will include investigations of curious design agents applied to more complex design tasks. For example, a natural progression is to apply curious design agents to the design of large public spaces like train stations where frequent interactions between pedestrians and the resulting emergent group behaviours have a significant impact on the performance of the space.

The ultimate goal of this work is to develop design agents that can assist an architect explore the issues involved in complex design tasks. Architects increasingly face the problem of "information overload" as they try to explore complex design spaces for innovative solutions. Although generative design tools relieve some of the burden of designing, they can make the problem of information overload worse as designers attempt to understand the significance of the designs produced. Technologies similar to curious design agents may play an important role in future CAAD systems by reducing the number of designs presented to an architect to a subset of those that are judged to be potentially interesting.

Providing design agents with motivations that reward the discovery of interesting designs more closely matches the motivations behind human exploration of design spaces. The application of curious design agents may allow future CAAD systems to provide more natural and rewarding collaborative partnerships between designer and machine.

5. ACKNOWLEDGEMENTS

This research is supported by an Overseas Postgraduate Research Scholarship and by a University of Sydney Postgraduate Award. We wish to thank Hsien-Hui (Michael) Tang for his valuable "non-computational" insights.

6. REFERENCES

Berlyne, D. E., 1971, *Aesthetics and Psychobiology*, Appleton-Century-Crofts, New York.

Damski, J. C. and Gero J. S., 1996, "A logic-based framework for shape representation", *Computer-Aided Design*, 28(3): 169–181.

Edmonds, E. and Soufi, B., 1992, "The computational modelling of emergent shapes in design", in: Gero and Sudweeks, (eds.), Computational Models of Creative Design, The Key Centre of Design Computing, The University of Sydney, Sydney, p.173–189.

Gaver, W. W. and Mandler, G., 1987, "Play it again, Sam: On liking music", *Cognition and Emotion*, 1(3): 259–282.

Gero, J. S., 1994a, "Computational models of creative design processes", in: Dartnall (ed.) *Artificial Intelligence and Creativity*, Kluwer, Dordrecht, p. 269–281.

Gero, J. S., 1994b, "Towards a model of exploration in computer-aided design", in: Gero and Tyugu (eds.), *Formal Design Methods for Computer-Aided Design*, North-Holland, Amsterdam, p. 271–291.

Gero, J. S. and Jun, H. J., 1995, "Visual semantics emergence to support creative designing: a computational view", in: Gero, Maher, and Sudweeks (eds.), *Preprints Computational Models of Creative Design*, Key Centre of Design Computing, The University of Sydney, Sydney, p. 87–116

Gero, J. S. and Saunders, R., 2000, "Constructed representations and their functions in computational models of designing", in: Tang, Tan and Wong (eds.), *CAADRIA 2000*, CASA, Singapore, p. 215-224.

Gero, J. S. and Yan, M., 1993, "Discovering emergent shapes using a data-driven symbolic model", in: Flemming and Van Wyk, (eds.), *CAAD Futures '93*, North Holland, Amsterdam, p. 3–17.

Helbing, D., 1991, "A mathematical model for the behavior of pedestrians", *Behavioral Science*, 36: 298–310.

Helbing, D. and Molnár, P., 1995, "Social force model for pedestrian dynamics", *Physical Review E*, 51: 4282–4286.

Helbing, D. and Molnár, P., 1997, "Self-organization phenomena in pedestrian crowds", in: Schweitzer (ed.) *Self-Organization of Complex Structures: From Individual to Collective Dynamics*, Gordon and Breach, London, p. 569–577.

Kohonen, T., 1993, *Self-Organization and Associative Memory*, 3rd ed., Springer, Berlin.

Liu, Y–T., 1993, "A connectionist approach to shape recognition and transformation", in: Flemming and Van Wyk, (eds), *CAAD Futures '93*, North Holland, Amsterdam, p. 19–36.

Major, M. D., Stonor, T. and Penn, A., 1998, "Passengers, pedestrians and shoppers: Space syntax in design", *Passenger Terminal World*, April.

Marsland, S., Nehmzow, U. and Shapiro, J., 2000, "A real–time novelty detector for a mobile robot", in *Proceedings of EUREL European Advanced Robotics Systems Conference*, Salford, England.

Martindale, C., 1990, *The Clockwork Muse*, Basic Books, New York.

Mitchell, W. J., 1993, "A computational view of design creativity", in: Gero and Maher (eds.) *Modelling Creativity and Knowledge-Based Creative Design*, Lawrence Erlbaum, p. 25–42.

Purcell, T. A. and Gero, J. S., 1992, "Effects of examples on the results of a design activity", *Knowledge-Based Systems*, 5(1): 82–91.

Schön, D. A. and Wiggins, G., 1992, "Kinds of seeing and their functions in designing", *Design Studies*, 13: 135–156.

Schmidhuber, J., 1991, "A possibility for implementing curiosity and boredom in model-building neural controllers", in: Meyer and Wilson (eds.) *Proceedings of the International Conference on Simulation of Adaptive Behaviour: From Animals to Animats*, MIT Press/Bradford Books, Cambridge, MA, p. 222–227.

Schmidhuber, J., 1997, "What's interesting?" *Technical Report TR-35-97*, IDSIA, Lugano, Switzerland.

Suwa, M., Gero, J. S. and Purcell, T., 1999, "Unexpected discoveries and S-invention of design requirements: A key to creative designs", in Gero and Maher (eds.), *Computational Models of Creative Design IV*, Key Centre of Design Computing and Cognition, The University of Sydney, Sydney, p. 297–320.

Tarassenko, L., Hayton, P., Cerneaz, N. and Brady, M., 1995, "Novelty detection for the identification of masses in mammograms", in: *Fourth International Conference on Artificial Neural Networks*, Cambridge, p. 442–447.

Tomlinson, P. and Gero, J. S., 1997, "Emergent shape generation via the boundary contour system", in: Junge (ed.), *CAAD Futures 1997*, Kluwer, Dordrecht, p. 865–874.

Whitfield, A. and Wiltshire, J., 1982, "Design training and aesthetic evaluation: An intergroup comparison", *Journal of Environmental Psychology*, 2: 109–117.

Worden. K., 1997, "Structural fault detection using a novelty measure", *Journal of Sound and Vibration*, 201(1): 85–101.

Yang, Y., Pierce, T. and Carbonell, J. G., 1998, "A study on retrospective and on-line event detection", *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'98)*, p. 28–36.

Moving Architecture and Transiting Landscape

Interactive Rendering System for Animated Assessment

Lee W.K. Alpha and Kazuhisa Iki
Graduate School of Science and Technology, Kumamoto University, Japan

Key words: Collaborative Design, Utilization of Internet, Overall Design Strategy,

Abstract: In this paper, an Interactive Rendering System for Animated Assessment (IRSA2) is proposed. Using IRSA2, different to the usual process that the respondents are allowed only to select alternatives designed by planners, they are allowed to participate in the design process and create alternatives as proposals in a web-based collaborative environment. This gives roads to an autonomous process in landscape planning and design. The system efficiency was verified by a case study of its use in a wind farm project in Japan.

1. PROLOGUE

If we clip landscape as "Scene Landscape" that resembles photos taken from any section of time, there is no transformation or variation. However, before one's very eyes, even if landscape is viewed from the same viewpoint in the same direction, it comes in difference faces. This change of landscape can be conceived and classified broadly into "Transformation due to Fluctuating Factors" such as time, season and climate, and "Transition due to the Change of the Subject" in the passage of a long period of time. When we talk about the change of landscape due to time, in addition to the sky color and the luminous intensity, the transformation of the shadow and change in the level of refection due to the change of the direction and position of the sun has to be considered. Especially at night, in addition to the natural light sources such as the moon and the stars, artificial light sources are important factors to be considered. For the seasonal changes, the introduction of trees and flowers, which embody the seasonal change, in landscape design is necessary. And for the climatic changes, the lowering of visibility due to rain

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 739–752.
© 2001 *Kluwer Academic Publishers.*

and fog, the change of sun condition due to movement of cloud by wind, and the covering of the subject by snow, visually change our perception of landscape. Besides, by including objects in motion, literally termed "Dynamic Landscape", in the framework of landscape, the landscape is becoming intimate and natural.

1.1 Objective of Study

Attempts to visually express the above "Transforming and Dynamic" faces of landscape by computer animations are successful. However, since animations are produced usually for viewing purpose according to the plots of the producer, it was not possible to change the condition of animation interactively by the observant. This presentation-oriented use of animation is not suitable for interactive assessment of landscape variables such as colour, textures that are too numerous to handle. Generally, perspectives, models, and recently CG are used to predict the result of architecture and landscape. Alternatives are made till a desirable solution is attained. In large-scale project, especially in public design projects that government sector and many specialists are involved, the above design process is very time consuming and thus costly. To response to the above problems, an overall strategic system to support the "Interactive Manipulation of Animated Landscape Assessment Constituents" is necessary.

Past studies of animation using information technology can be classified into 2 main categories, namely "Simulation of Architecture or Landscape for Assessment", and "Evaluation of the Reality and Cost performance of Simulation Techniques". The former is concerned mainly with the use of animation in the design and working stage (Mc Cartney, et al., 1997) of architectural (Ai, Tang, et al., 2000) or landscape planning (Clayden, Szalapaj, 1997), and the latter is concerned mainly with the quality (Bermudez, et al., 1998) and cost performance (Burry, 1997) of the display media. As a continuation to the authors' study on the utilization of Internet in interactive interdisciplinary design, the possibility of the latest computer game supporting technology and web technology to support the "Interactive Manipulation of Animated Landscape Assessment Constituents" in architectural and landscape design is examined. In this paper, an internet-based system, IRSA2 (Interactive Rendering System for Animated Assessment) is proposed. Different to the usual process that the respondents are allowed only to select alternatives designed by planners, they are allowed to participate in the design process by interactively manipulating the landscape assessment constituents and create alternatives as proposals.

1.2 Method of Study

As shown in figure 1, in regard to IRSA2's use in Internet, web-based Interactive Landscape Stimulation Supporting Tools are included to response to the main criteria of "Visual and Corporal Assessment" of "Scene and Sequence Landscape" in the stage of Object Design (namely "Nature of Objects", "Relation of Object and Viewpoints", "Mutual Relationship of Objects" and "Change of Landscape"). Web-based Interactive Data Management & Display Supporting Tools are included to response to the human perceptibility, share of information and the collaborative activities in landscape planning. To verify the efficiency of the IRSA2, as case study, attitude survey for the color planning of a wind farm was undertaken.

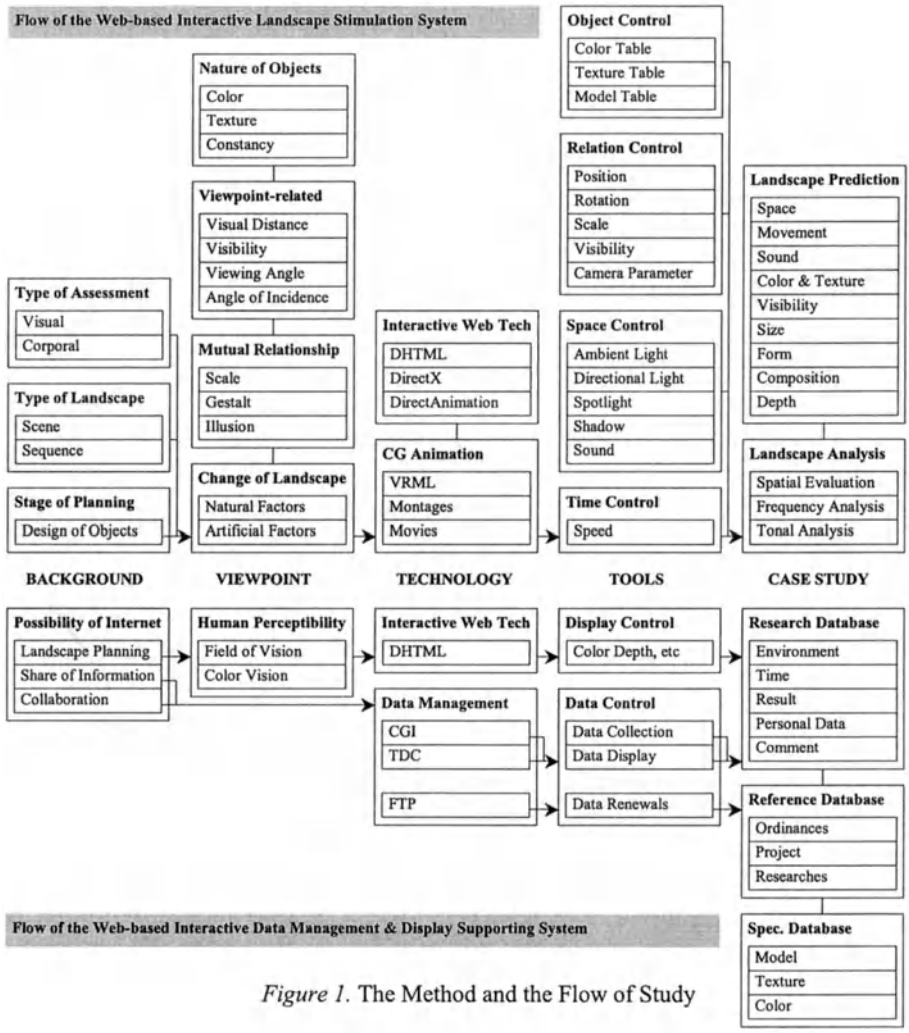


Figure 1. The Method and the Flow of Study

2. CONSTRUCTION OF SYSTEM

IRSA2 is constructed in a multimedia platform of DirectX technology, with the integration of Structured Graphics Control of DirectAnimation, interactivity of DHTML (Dynamic HyperText Macro Language), TDC (Tabular Data Control), CGI (Common Gateway Interface) and FTP (File Transfer Protocol). With the insertion of montages, movies and VRML models, "Moving Architecture and Transiting Landscape" can be simulated. Real time interactive movement and rotation of model parts, instantaneous change of environmental factors such as light source, display of shadow and setting of audio effects, sorting and filtering of database, etc. are possible.

2.1 Specification

As shown in figure 1, in regard to the Reality, Precision, Extendibility, Manoeuvrability and Cost of Simulation, the following specification are considered in the construction of IRSA2.

2.1.1 Interactive Data Management & Display Supporting Tools

a) Artificial Intelligent: Evaluation of display suitability (color depth and screen proportion, etc.) by DHTML.
b) Pool Style: Collection of environment parameters (operating system, duration of assessment, participant's personal details, comments, etc.) and selections of alternatives (color and texture used, camera parameters, etc.) by TDC.
c) Library Style: Synchronous sorting and referencing of data by TDC. Renewals of reference and specification database by FTP.

2.1.2 Interactive Landscape Simulation Supporting Tools

a) Adjustment of animation speed and visibility (Kawasaki, 1998), and change of background (movies, geographical models) by DHTML.
b) Real time color simulation, texture mapping and change of opacity of models for change of the "Nature of Objects" by DHTML
c) Real time simulation of visual distance, viewing angle, scale, etc for adjustment of "Mutual Relationship of Objects" and "Relation of Object and Viewpoints" by DirectAnimation.
d) Real time simulation of environmental parameters such as directional light, ambient light, spotlight, shadows and sounds by the integration DHTML and DirectAnimation.

2.2 Structure of IRSA2

Landscape Desktop is an interface for the application of the user-friendly IRSA2. As shown in figure 2, Landscape Desktop consists of 2 parts, internal and external. For the control of interactive simulation and database display, collection and management, Landscape Desktop is composed of 5 parts, namely, Dialogue Box (System, Palette, Texture, Model, Cost), Rendering Frame, Pop-up Frame, Side Frame and Information Display Tools. Interactive tools formerly devised by the authors, ITAS (Lee, Iki, 2000a), ICPS (Lee, Iki, et al., 2000b), IRSCD (Lee, Iki, et al., 2000c), IRSC3D (Lee, Iki, 2000d), IRSCP (Lee, Iki, et al., 2000e) are included.

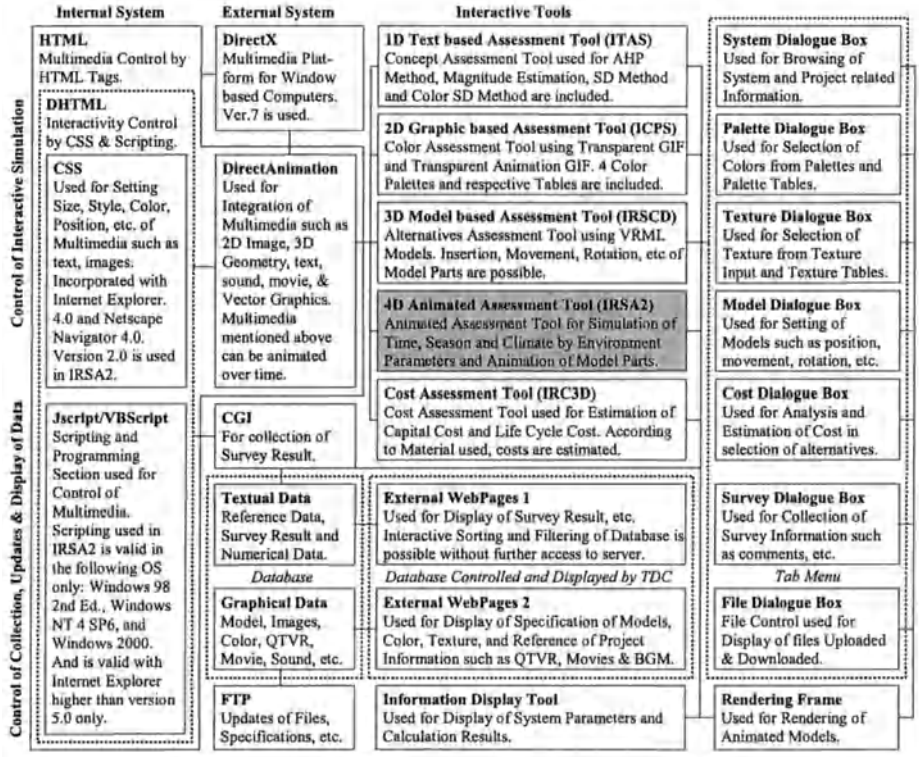


Figure 2. Structure of Landscape Desktop and the Position of IRSA2

2.3 Interactive Web Technology

IRSA2 is constructed using the following interactive web technologies with scripts compatible for use in Windows 2000 and Windows 98 2nd edition, with Internet Explorer 5.5 as the browsing environment. In addition, CGI is used for data collection and FTP is used for data updates.

The various interactive functions of IRSA2 during the result evaluation mode of IRSA2 are shown in figure 3.

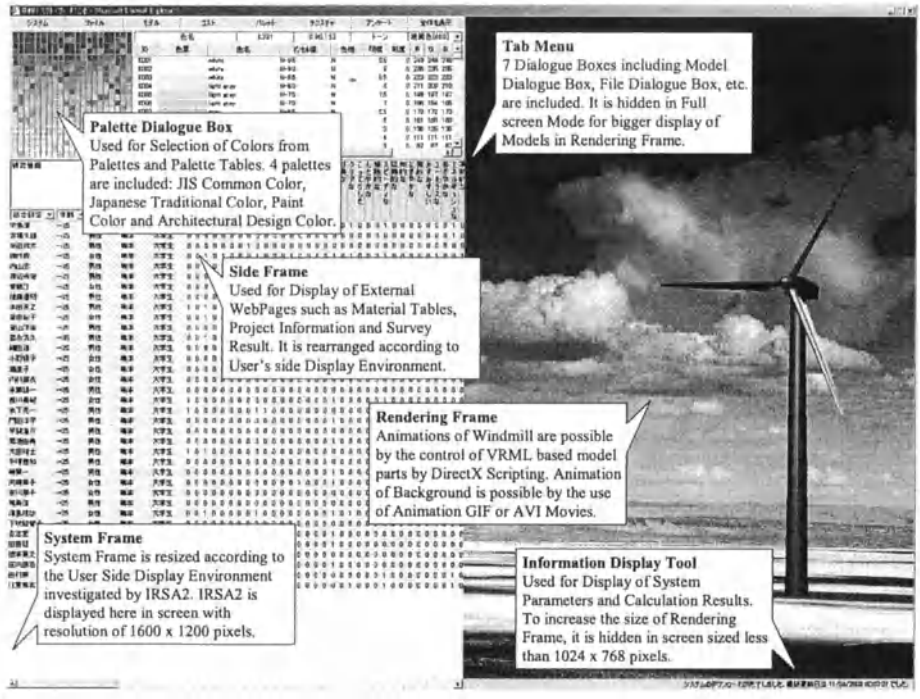


Figure 3. Result Evaluation Mode of IRSA2

2.3.1 DirectAnimation

DirectX is a family of high-level multimedia controls and application program interface. DirectAnimation is the DirectX component that provides support for animation, streaming, and integration of different multimedia. DirectX version 7 and DirectAnimation version 6 are used for rendering and animation of models in IRSA2.

2.3.2 DHTML

DHTML is an extension to HTML. Interactive WebPages can be constructed with the present HTML tags, by inclusion of CSS (Cascading Style Sheets) and scripting such as JavaScript Version 1.2 and VBScript. As shown in Figure 4, since all contents of WebPages are downloaded all at once, reloading is not necessary to response to user's demand.

2.3.3 TDC

TDC is an extension of DHTML. It allows the display of textual and multimedia database in tabular format. As shown in Figure 5, once downloaded, the database can be sorted and filtered at user's side without further access to the server and the load of server can be cut significantly.

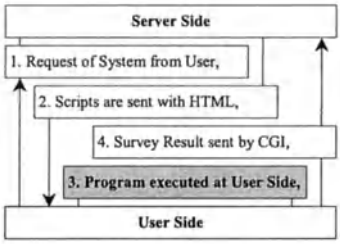


Figure 4. DHTML

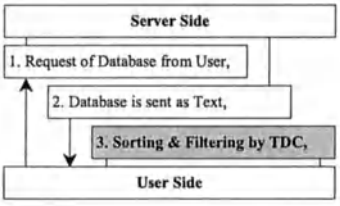


Figure 5. TDC

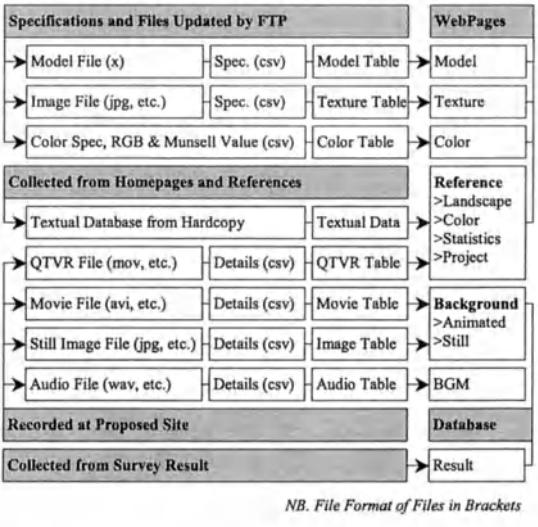


Figure 6. Mechanism of External WebPages & Database

2.4 External Databases and WebPages

Figure 6 is the mechanism of external database and WebPages. They are used for reference and simulation during assessment. Figure 7 is an example of external WebPages of color specifications.

2.4.1 Project Database

The following 3 types of project database are included. They are recorded at the proposed site.

a) QTVR Files: for reference of site environment.
b) Movie Files: as animated background image.
c) Audio Files: as audio effects.

2.4.2 Reference Database

The following 6 types of reference database are collected for reference.

a) Ordinances related to landscape and townscape planning.

b) Color-related researches.
c) Statistics.
d) Related project.
e) System data.
f) Reports and Papers.

2.4.3 Material and Model Database

The following 3 types of material and model database are made for simulation and rendering.
a) Model libraries and model tables
b) Texture libraries and texture tables
c) Color palettes and color tables

2.4.4 Assessment Result Database

Environment parameters and model settings together with personal information, comments are collected by CGI and saved in the assessment result database for further analysis. The database is in numerical and textural format that can be displayed through TDC.

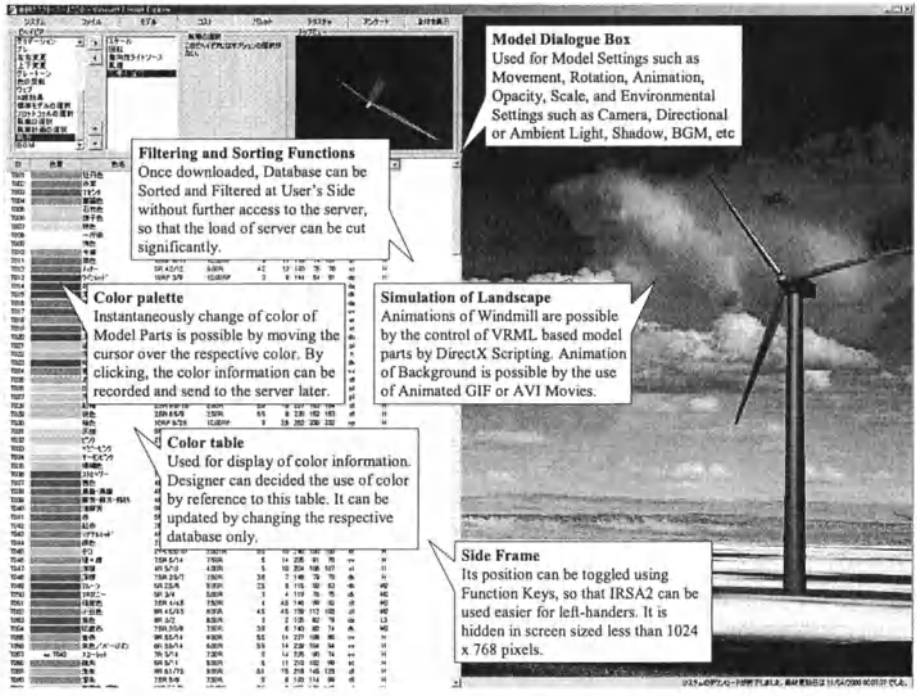


Figure 7. External Databases and WebPages in Color Evaluation Mode of IRSA2

3. CASE STUDY

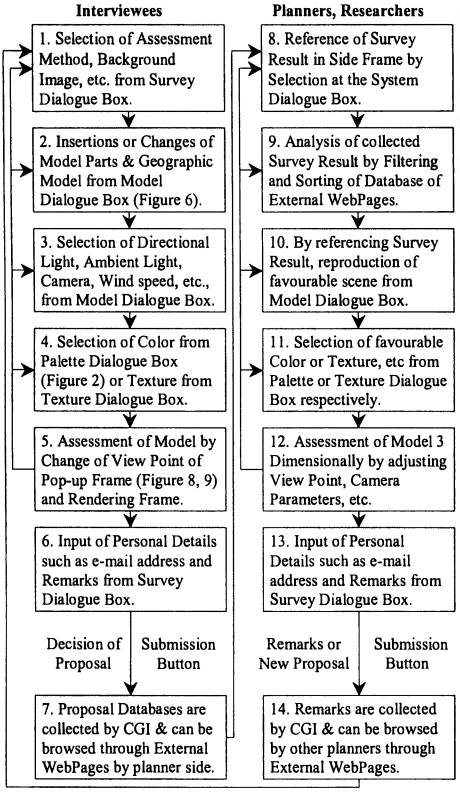


Figure 8. Flow of Case Study

Figure 9. Still Image & Top View Pop-up

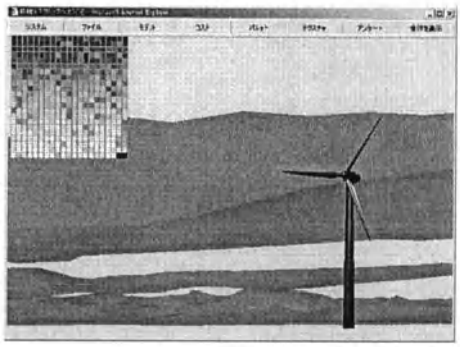


Figure 10. Geographic Model & Color Pop-up

3.1 Selection of Subject of Assessment

Propeller windmills, being high in energy conversion performance, are usually used in wind farm projects. They are classified generally by size into the following 4 categories: Micro (diameter < 3m), Small (3m < diameter < 20m), Middle-sized (20m < diameter < 45m) and Big (diameter over 45m). In introduction of middle-sized and big windmills, for high-energy conversion performance and for convenience of repairing, windmills are usually placed in areas with no wind-breaking obstacles and in areas with good road access. In addition to the problem of noise, being big in size, their visible territories are so large that inevitably leads to the detrimental effect on natural landscape. As a result, it is necessary to devise methods to minimize their damages to the landscape and evaluations from interested parties are necessary before construction. In this research, as an example of "Moving Architecture", using IRSA2, a VRML model of a 76m high propeller windmills is selected as the subject of assessment in the color planning of a windmill in Japan.

3.2 Flow of Case Study

The flow of case study using IRSA2 is shown in figure 8. Regardless of the limitation of time and space, respondents, researchers and planners, using a common browser, in the same platform, share the same information for landscape assessment. Respondents can assess the model by changing color, texture and size of the models and evaluate the result in different viewpoints by changing the camera settings. The above process can be repeated till a suitable solution is attained. The researchers and planners can reference the result collected for feedback in the actual planning purpose. Favourable design can be re-simulated by inputting the relevant parameters used during the assessment.

3.3 Details of Case Study

The VRML model of a propeller windmill was constructed for assessment purpose. To consider the influence of moving wind-blade of the propeller windmill in the assessment process, the rotation of wind-blade is animated by adjustment of the rotation velocity according to the setting of wind speed parameters. 32 university student specialized in architecture are requested for the assessment. To consider the actual application of color, a paint color palette of 343 colors (JPMA, 1999) is constructed for the assessment.

Using IRSA2, to relieve the psychological stress, respondents are requested to choose UNFAVOURABLE color of the propeller windmill with respect to the background environment (sky and mountain), and restrain from choosing favourable color because of one's personal preference. To grasp the band of individual difference, number of color chosen is not fixed and the assessment time is limited to approximately 20 minutes. The following is an explanation of the assessment environment: 1. Operating System: Windows 2000, 2. Browser: Microsoft Internet Explorer version 5.5, 3. Available Width of Screen: 1600 pixels, 4. Available Height of Screen: 1172 pixels, 5. Color Depth of Screen: 32 bits.

3.4 Overall Analysis

Since the number of color chosen is of wide band (33 to 339 colors), it is necessary to consider the personal difference (average number of color chosen: 204). In this research, analysis based on the Average Deviation (absolute deviation of data in respect to the average of the same group, an indicator of the dispersion of the data) of color chosen in respective Hue and Tone are undertaken. The results are as follow.

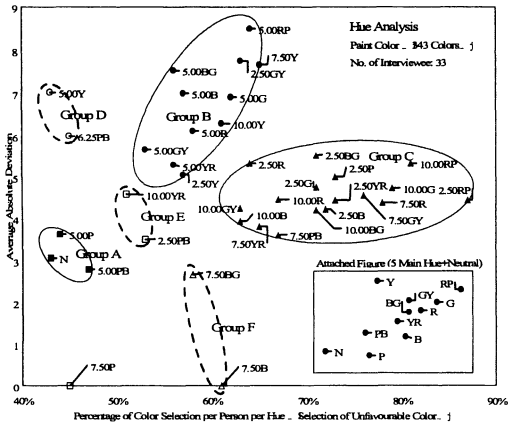


Figure 11. Result of Hue

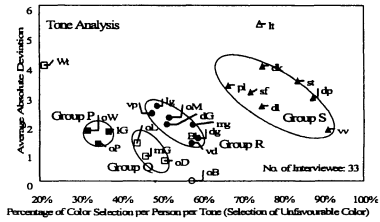


Figure 12. Result of Tone

3.4.1 Result of Hue

As shown in the attached figure in figure 11, being low in personal difference and least selected as unfavourable color, the most favourable Hue is Neutral (N). Being high in personal difference and highly selected as unfavourable color, it is advisable not to use Red Purple (RP) in the planning of windmill. As shown in figure 11, according to the dispersion, the colors are grouped in 6 different Hue Groups (A, B, C, D, E and F) for analysis. Being low in personal difference and least selected as unfavourable color, Group A is considered as the most favourable group. And with the same reason, Group E and Group F are considered as the candidate groups. Being high in personal difference, the use of Group B and Group D should be with attention. Being least selected as unfavourable color, it is advisable not to use Group C.

3.4.2 Result of Tone

As shown in figure 12, based on ABC Method used for Paint Color Tone Classification in Japan, the colors are divided in 24 groups for analysis. Though least selected as unfavourable color, being high in personal difference, the use of White (Wt) should be with attention. With high personal difference and being highly selected as unfavourable color, the use light (lt) is not advisable. According to the dispersion, the colors are grouped into 5 main groups (P, Q, R and S) for analysis. Being low in personal difference and least selected as unfavourable color, Group P is considered as the most favourable group and Group Q and R are considered as the candidate groups. Being highly selected as unfavourable color, the use of Group S is not advisable.

4. EFFICIENCY OF IRSA2

The system efficiency is verified under the following 5 viewpoints.

4.1 Reality

a) Environment quality such as angle of incident of the sun, vanishing point, and model characteristics such as texture can be simulated.
b) Movies and still images used as background are photographed at the eye-level of an average 20-year-old Japanese man with 50mm lens.
c) Geographic models are made with 50m meshes. With texture mapping, realistic rendering is possible.
d) By adjustment of the level of ambient light and the 3 axial directions and angles of directional light, the change of time can be simulated.
e) By adjustment of the above light settings, and the use of movies and still images as background, the seasonal changes can be simulated.
f) By animating the wind blade rotation of windmill and the use of movie as background, the simulation of wind is possible.
g) By adjustment of the visibility of the environment, the simulation of climatic change such as fog and rain is possible.
h) By using audio clips collected at site, the simulation of realistic acoustic environment is possible.

4.2 Precision

a) Models made from working drawings by CAD are converted to VRML models. High precision is possible regardless of the scale.
b) In regard to its use in Internet, the tolerance due to the difference of color space definition in Munsell-RGB conversion is considered acceptable (Newman, 1995). Since the range of human color perceptions is 6 to 7 million colors, with the use of high-resolution monitors, simulation of high precision is possible.

4.3 Extendibility

a) In regard to its use in Internet, the file size of the system is limited to 50 KB. The system is extendible through the use of Landscape Desktop.
b) Incorporation with other multimedia contents such as Chat and Net Conferencing is possible using Direct Play.
c) Color and texture tables and model parts can be incorporated as extension, according to the type of assessment.

d) By extension, the use in other field of design such as interior design, product design is possible.

4.4 Manoeuvrability

a) By the use of TDC, interactive sorting and filtering of database is possible without further access to server once downloaded.
b) Instantaneous change of model characteristics and environmental Settings are possible during animated assessment in the same screen.
c) Since the subject of assessment is considered as an assembly of model parts, insertion, deletion and updates of model parts are possible.
d) By the use of FTP, sharing of database by updates of model parts, texture images, color specifications, etc is possible.
e) System frames are rearranged and resized according to the user-side display environment and operating system investigated by IRSA2.

4.5 Cost

a) Being application-independent, unnecessary software updates for data management software for example can be cut.
b) Reuse of databases such model parts, color specifications, texture Images and project information saved in server libraries is possible.
c) Survey results are collected as numerical and textual data in tabular format that can be used for evaluation and analysis directly.

5. CONCLUSION

With the use of IRSA2, "Interactive Assessment of Moving Architecture and Transiting Landscape" is possible with instantaneous simulation of high reality during change of the various landscape elements, independent of time and space. The favourableness of each element and the band of individual difference can be grasped for further reference in landscape planning. In the above process, different to the usual process that the respondents are allowed only to select alternatives designed by planners, respondents are allowed to participate in the design process and create alternatives as proposals, which gives roads to an "Autonomous Process" in landscape planning. And since the systems proposed are internet-based, they can be used in Web and in standalone basis. Even without connection to server, it can be used for insitu investigation and education purpose, which is effective for citizen-participation-oriented landscape planning. As a future prospect, with the improvement of transfer speed of Internet and polygon rendering capacity of

the browser environment, the incorporation of other element of transiting landscape, such as animals, people, cars are also in consideration and the reality of the seasonal changes can be improved.

6. REFERENCES

Ai, J., H. Tang, and Y. Chen, 2000, "An Approach to Generate 3D Animation by Integrating Building Model into Site Pictures", *5th CAADRIA Conference Proceedings*, p. 433-439.

Bermudez, J., et al., 1998, "Media Interaction & Design Process: Establishing a Knowledge Base, Digital Design Studios: Do Computers Make a Difference?", *3rd CAADRIA Conference Proceedings*, p. 6-25.

Brady, Darlene A., 1997, "The Mind's Eye: Movement and Time in Architecture, Design and Representation", *ACADIA '97 Conference Proceedings*, p.85-93.

Burry, M., 1997, "The Cost and Value of Animation for Architectural Designers", *1st AVOCAAD Conference Proceedings*, p. 135-144.

Clayden, A. and P. Szalapaj, 1997, "Architecture in Landscape: Integrated CAD Environments for Contextually Situated Design, Challenges of the Future", *15th eCAADe Conference Proceedings.*

Japan Paint Manufacturers Association (JPMA), 1999, *Standard Paint Colors*, JPMA, Tokyo.

Kawasaki Y., 1998, "Study on a Method of Representation of Color Perspective using Computer Graphics – Representation of Color Perspective using Filter Operation", *Journal of Architecture, Planning and Environmental Engineering*, Architectural Institute of Japan, Vol.511, p. 153-159.

Lee, A., K. Iki, 2000a, "Use of DHTML For Interactive Assessment of Common Value For Townscape Conceptualization and Realization - Color Assessment, Case Study of Large-Scale Resort Facility in Aso Region, Kumamoto Prefecture, Japan", *Proceedings of the 5th Conference on Computer-Aided Architectural Design Research in Asia*, p. 89-96.

Lee, A., K. Iki, M. Morozumi, 2000b, "Interactive Color Planning System", *Journal of Architecture, Planning and Environmental Engineering*, Architectural Institute of Japan, no.537, p. 325-332.

Lee, A., K. Iki, M. Morozumi, 2000c, "Interactive Rendering System for Collaborative Design", *Journal of Architecture, Planning and Environmental Engineering*, Architectural Institute of Japan, no.538, p. 291-297.

Lee, A., K. Iki, 2000d, "Interactive Rendering System for Cost Conscious Collaborative Design", *Proceedings of the 3rd International Conference on Computer-Aided Industrial Design and Computer-Aided Conceptual Design.*

Lee, A., K. Iki, M. Morozumi, 2000e, "Interactive Rendering System for Cost Planning, the use in Early Stage of Working Design", *Journal of Architecture, Planning and Environmental Engineering*, Architectural Institute of Japan, no.543.

Lyons, A. and C. Doidge, 1993, "Understanding Structural Movement Joints with CAAD Animation, *11th eCAADe Conference Proceedings.*

Mc Cartney, et al., 1997, "Testing the Benefits of Animation", *1st AVOCAAD Conference Proceedings*, p. 255-262.

Newman, T., 1995, "Improved Color for the World Wide Web - A Case Study in Color Management for Distributed Digital Media", *Proceedings of the TAGA*, Vol.2, p. 772-787.

Interactive 3D reconstruction for urban areas

An image based tool

C. Chevrier and J.P. Perrin
CRAI UMR MAP 694, School of Architecture of Nancy

Key words: geometrical modelling, architecture, urban area, virtual visit.

Abstract: Urban applications (for example arrangement, new buildings, virtual sightseeing and walkthrough) require a three dimensional (3D) geometrical model of town areas. However, most of them do not need an accurate model of reality. Such model would occupy a considerable memory space and would be too slow to handle. Architects, urban designers and civil engineers can find in our tool a medium to conceive their projects. Some types of software exist but they do not correspond exactly to our needs. Consequently we have conceived and developed an interactive tool for virtual 3D rough reconstruction of buildings. The software development has been performed in the Maya environment (ALIAS Wavefront) with C++ language and MEL (Maya Embedded Language). A constraint we set for ourselves was the use of only light devices (for easy transportation) at low price (everybody can buy such devices). The principle is to overlay on the scanned photograph of the area we want to deal with, the two dimensional (2D) cadastral plan displayed from the same viewpoint as the picture. Then each building body can be extruded from its ground polygon and the roof can be created from what the user sees on the picture. A constraint is the flatness of the polygonal surfaces. Our application context was the town of Nancy in France for which some areas have been reconstructed. Some pictures have been used as textures for polygonal surfaces, giving more reality effect to the simulation.

1. INTRODUCTION

Urban applications require a three dimensional (3D) geometrical model of town areas: urban arrangements, new buildings (augmented reality simulations with handling of the interactions between real and virtual worlds (Chevrier, 1996b), virtual sightseeing and walkthrough). However, most of

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 753–765.
© 2001 *Kluwer Academic Publishers.*

We have conceived and developed an interactive tool for virtual 3D rough reconstruction of buildings. The software development has been performed in the Maya environment (ALIAS Wavefront). Architects, urban designers and civil engineers can find in this tool a medium to conceive their projects.

The principle is to overlay on the scanned photograph of the area we want to deal with, the two dimensional (2D) cadastral plan displayed from the same viewpoint as the picture. Then each building body can be extruded from its ground polygon and the roof can be created from what the user sees on the picture. A constraint is the flatness of the polygonal surfaces.

We developed a set of commands making the creation of various roof shapes easier, taking advantage of the use of photographs. A constraint we also set for ourselves was the use of only light devices at low price: only a camera without any specific features (simpler if it is a digital camera) and if possible a hand-held distance meter are required. This lowers the cost of the devices and the time spent for the camera shots. We can use as input data, the results of the automatic rough reconstruction from Medina software (Allani and Perrin, 1998) developed by our team.

Part 2 presents the state of the art for 3D reconstruction and viewpoint recovery. The next Part (3) explains the principles of our method and the required input data. Then, we see in the following part (4) how to retrieve the photograph viewpoints. The reconstruction task can finally begin with the extrusion of the building bodies (part 5) and the building up of the roofs (part 6). Part 7 presents the results with the help of a concrete application. Finally, part 8 presents future work and concludes.

2. STATE OF THE ART

Medina (Allani and Perrin, 1998) is a program for automatic 3D simplified reconstruction from 2D cadastral plans. It utilises information stored in the plans (for instance the number of storeys in a Dxf format file), urban regulations and architectural laws to build up the global shape of the urban area being treated. The most appropriate roof is constructed for each building. This program allows one quick reconstruction of large areas. However, the results lack accuracy for some applications (augmented reality) and complex roofs are not correctly managed. Nevertheless, the resulting file from Medina can be used as an input for our module. Only the badly or non built-up houses are then dealt with.

Different types of commercial interactive software exist to solve the problem of 3D reconstruction. Let us examine some of them. Canoma (Canoma, 2000), ImageModeler (RealViz, 2000) and PhotoModeler

(PhotoModeler, 2000) are software based on the principle of photograph aided modelling.

PhotoModeler is based on the principles of photogrammetry. Canoma does not require a viewpoint recovery of the digital images, but several images of the same object have to be used in order to fix the object position and shape correctly in the scene. On the contrary, ImageModeler and PhotoModeler require a viewpoint recovery. The calibration process enables the building in the scene of the 3D indices corresponding to the 2D indices selected on the pictures. Then, the user can rebuild the objects with the help of these 3D indices and the photographs. In the case of PhotoModeler, the 2D indices can be segments (not only points); one can automatically obtain a 3D model composed of segments. All these software have simple primitives such as plane, cube, cylinder, cone, and sphere, interactively positioned in the scene with manipulators. Main shapes for a rough reconstruction are more or less provided according to the software. However, no composition of simple roofs to create complex roofing is possible. An application example of Canoma is the reconstruction of parts of the town of Phoenix (Arizona) mainly composed of box-shaped buildings. Nothing ensures the flatness of the polygons after modification in most programs. In architecture, most of the roofs are composed of planar polygons: it is important to respect this constraint. These software are not based on modellers so they do not dispose of classical commands. They can render images by wrapping the photographs around the 3D primitives.

Automatic reconstruction with the help of image analysis process produces good results for indoor and simple scenes. For urban scenes, most of the research carried out on that subject uses aerial video sequences (Faugeras, laveau, et al., 1995) (Collins, Hanson, et al., 1995). Photographs taken by a walking man are difficult to deal with: scenes are composed of lots of objects of various kinds (lots of "parasite" objects in front of the buildings).

We want to create an interactive tool for helping 3D rough reconstruction of urban areas. To simplify this task an interactive modeller is useful. Classical modellers (AutoCad, Arc+) allow neither intuitive and simple handling of the objects, nor real-time 3D visualisation. Maya contains several modules, among which such a modeller. The Open Inventor library (Wernecke, 1995) allows real time visualisation and handling with manipulators of 3D graphical objects. However, no standard modelling feature is provided. Thus, we choose to develop our prototype in the Maya environment with the C++ programming language and MEL (Maya Embedded Language), avoiding the fastidious task of writing all the standard operations of a modeller. Nevertheless, we can note that Maya's modeller is not as complete and accurate as classical modellers but it fits our needs.

Furthermore, Maya allows us to overlay a 3D scene on a background image. This image is associated to a given camera viewpoint. Several cameras and also several images can be used at the same time for our reconstruction goal.

As far as the viewpoint recovery is concerned, one can find more and more commercial software (realViz, 3Dstudio, Mayalive) and a large amount of research has been carried out on that subject (Devernay, Faugeras, 1995)(Berger, Chevrier, et al., 1996)(Simon, Berger, 1999). Some tests with MayaLive did not satisfy us for several reasons:

1) MayaLive deals with video sequences and not with just one picture. It obliged us to film the scene instead of taking a few pictures. We had to use a tripod and to borrow an expensive digital video camera. The room taken up by such devices was too great and they were not easy to move. The time spent on the shooting was also long.

2) The recovery of the viewpoint requires several steps (as usual for video sequences, whatever the software): 2D relevant indices tracking and resolution of the system. These steps are too long for the one picture that interests us in the sequence.

3) The first tests we made were in an indoor courtyard of the school of Architecture (a very linear new building). The results were very promising. Unfortunately, trials in an urban context were not satisfactory: we do not dispose of enough relevant indices in the images and in the 3D scene. Points in the cadastral plan were often not visible in the images because they were hidden by other objects (dustbins, cars, trees, low walls, etc).

ImageModeler and PhotoModeler utilise calibrated images for the 3D reconstruction. One can use pictures or movies. Camera calibration is processed as in MayaLive with the help of relevant indices (points in ImageModeler, points and segments in PhotoModeler) seen on at least three images. These points are manually pointed with a large cover in the images and in the three main directions of the space. Six points are required. 3D calculated points can be manually adjusted in case of errors and calibration is processed subsequently. In our applications, we mainly have at our disposal points on a 2D plan (cadastral plan), and hardly any indices in the vertical direction.

Thus we have developed a simple, interactive and sufficiently effective method for our needs in order to recover the viewpoint of a scanned photograph. This method was created in Maya, ensuring compatibility with the 3D reconstruction software, as in the same environment.

3. PRINCIPLES

3.1 Steps of the reconstruction

The **first** step consists, for each picture, in recovering the camera position in the scene in order to visualise the 2D plan according to the same viewpoint. The **second** step is the creation of the building bodies: ground polygon extrusion (special case of the non-planar and non-horizontal ground). Finally the **third** step allows us to construct the roofs: Some commands make easier the creation of simple roofs (1, 2, 3 or 4 slopes, pyramidal or flat) and composed roofs (association of several simple roofs).

3.2 Input data

Geometrical data about the existing scenes are necessary for viewpoint recovery: this step utilises the matching of the 3D data and their corresponding 2D data on the images. The data we use are:

- a cadastral plan with ground point elevations. If we do not dispose of such altitudes, we can interpolate them from the altitudes of reference points (geodesic points). Cadastral data have sometimes to be cleaned and treated to create closed ground polygons.
- Coordinates of several points at other altitudes. These points are useful in order to recover the viewpoint. It reduces the risk of error in the third space direction. These points can be measured with the help of a hand laser: distance between the laser and a target point (for example the corner of the buildings at the roof gutter level). One point per picture is enough. If we do not dispose of such points, a small uncertainty will subsist on the height of the reconstructed objects. This will be minimised with the use of several images.
- Photographs of the area: pictures of a street-level pedestrian and heightened views shot from high buildings in order to have a better vision of the roofs. Images are taken with a digital camera thus avoiding losing time with development and scanning (scanning produces troubles with pixel size and position of the optical centre projection).

4. VIEWPOINT RECOVERY

Recovering the viewpoint implies identifying two sets of parameters: the intrinsic and the extrinsic parameters. This means eleven parameters in totality (Horaud, Monga, 1993)(Toscani, 1987).

Intrinsic parameters are specific to the camera and are unvarying when it moves in the scene: focal length or aperture, pixel size, position of the optical centre. The use of a calibration grid (regular grid) picture allows a reliable identification of these parameters (Toscani, 1987).

Extrinsic parameters are the position of the camera in the scene, the target and the rolling angle (orientation of the camera according to the sight axis). These parameters are specific to each image. If the intrinsic parameters have been previously determined, one can compute the extrinsic parameters from a small set of corresponding 2D and 3D relevant indices (Horaud, Monga, 1993). However, estimating these parameters for noisy and complex images is not easy (Chevrier, 1996a).

Trials carried out with MayaLive did not satisfy us, so we have developed a simple interactive method that allows us both quick and good positioning of the camera in the scene.

4.1 Principle

- We utilise a first fixed point ***I1*** chosen by the user in the image. To this point corresponds a 3D point ***P1*** in the scene. The two points are superimposed: translation of the camera (see Figure 1a).
- A second semi-fixed point ***I2*** is chosen in the image, to which corresponds the 3D point ***P2***. These two points are also stacked keeping the first superposition: rotation ***x*** around the X axis, rotation ***y*** around the transform of Y axis by rotation ***x***, scaling ***e*** of the image plane (see Figure 1b).
- The image plane can be rotated according to the two fixed points in order to correctly orientate it according to the scene: rotation ***r*** (see Figure 1c).
- The point ***P2*** can move along the line (Camera position ***C***, point ***I2*** on the image): parameter ***p***. A modification of ***p*** leads to a modification of scaling factor ***e*** and the rotation angles ***x*** and ***y*** (see Figure 1d).
- Finally the focal length ***f*** of the camera can vary. A modification of ***f*** leads to variations on the points ***I1*** and ***I2*** as the image plane dimensions change.

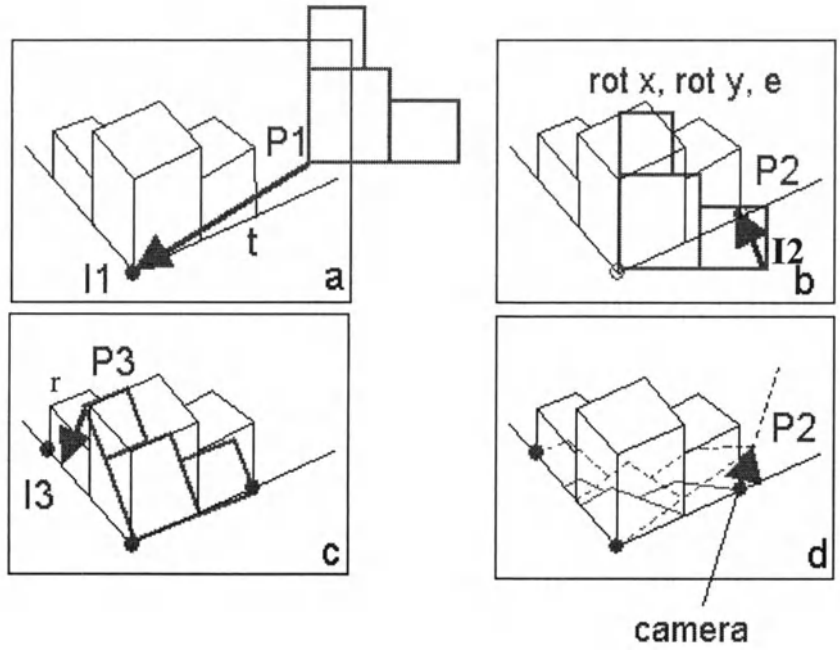


Figure 1. Estimation of the viewpoint

The first four parameters (***I1,I2,P1*** and ***P2***) are fixed once at the beginning by the user. The other three parameters (***p***, ***r*** and ***f***) must be adjusted together. From eleven unknown variables, we reduce that number to three. Furthermore, if the focal length is known, it can be fixed leaving us with only two unknown variables. In practice, even if the focal length has been fixed, we sometimes have to vary it a little to get better results. One can use the vanishing points to estimate the focal length value, and knowledge of the scene and the perspective of the image to approximately fix the starting value of parameter ***p***. Finally, one can approach the value of the rotation angle ***r*** according to the high angle shot or low angle shot.

Note that as we do not have at our disposal a lot of high points, we create some vertical lines from relevant points of the cadastral plan. When several buildings have been reconstructed, these lines are not useful any more: the buildings themselves are used.

In some cases, one can use a third fixed point ***I3***, corresponding to a 3D point ***P3***. This point ***P3*** must not be aligned with the two first points ***P1*** and ***P2***. This is not always possible, for instance when we have a picture of frontages in a street. Thanks to this third point, the rotation ***r*** can be

estimated from the value of the parameter ***p*** given by the user. With the rotation ***r***, the projection of ***P3*** on the image plane is on the line (***I1,I3***) (see Figure 1c). With a variation of the parameter ***p***, the projection of ***P3*** comes on the point ***I3***. This greatly simplifies the process. Only the parameter ***p*** needs to be estimated and the focal ***f*** sometimes needs to be adjusted.

4.2 Distorted images

Main distortions, which means radial distortions, are identical for all the points along a primary ray (ray shot from the eye towards a given direction). In image based aided work (medicine for example), it is often important to have non-distorted images but not necessary to know the shooting parameters. In this case, distorting the image is sufficient. It means putting it in conformity with a perfect perspective projection. In order to do that, a method of comparison between a perfect image and a photograph of a specific object (regular grid) allows one to globally correct all the distortions with no need to model them (Peuchot, 1994). As this kind of distortion depends on the focal length, this correction has to be done for each change in the focal length.

We use this method to distort our images before viewpoint recovery. For small focal lengths (less than 50 mm), one can observe distortions in a small cushion, and for long focal lengths (more than 50 mm), distortions are in a small barrel. The grid (composed of squares of 5cm) has been shot with our digital camera (Olympus) for three various focal lengths : the smallest, the greatest and one in the middle. The wide angle corresponds to a focal length of 9.2 (equivalent to 36 mm for a standard 24x36 camera). The telephoto focal length is 28mm (110 mm for a standard camera). We have developed a program to distort the images according to the focal length used. Most of the shots were made with the widest angle.

5. BUILDING BODIES

The second step is the extrusion of the ground polygons to create the building bodies. If the ground is not horizontal, we must first position the cadastral plan according to the ground altitudes: we vertically project the polygons on the 3D mesh created with the known-altitude points. With Maya, the extrusion of a polygon is performed perpendicularly to that polygon, which does not correspond to what we expected. Thus we develop a command to extrude vertically non-horizontal polygons: a bottom part is automatically created to get the horizontal level before extruding the polygon (Figure 2).

The extrusion can be performed with various options: a given height if an in-situ measure has been taken, or the number of storeys and a height per storey (estimated from architectural customs and laws of the construction period). Then, a manipulator can be used to adjust the height according to the picture.

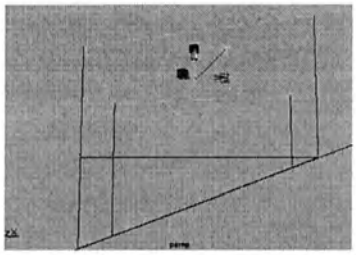


Figure 2. Polygon extrusion for a sloped ground

6. ROOF CONSTRUCTION

Thanks to the pictures, we are now able to create the roofs. We distinguish two types of roof: the simple roof and the composed roof. Simple roofs are pyramidal, flat, made of one, two, three or four slopes. Complex roofs are composed of several simple roofs with particular association rules: "A la Mansard" roofs, L, T or U shaped roofs.

A set of commands makes the roof creation easier taking into account the use of photographs. In classical modellers, roofs are created from the slope of the various sides. In our case, we do not know these data, so we have used other data: the position of the roof top. Two, three and four slope roofs are then created from the top.

6.1 Simple roofs

A flat roof is created by the raising of the top side points. The user can modify the position of the top points of the flat roof to create for example the bottom part of a "A la Mansard" roof. A simple roof (two or four slope roof) has to be constructed over that part to complete the roof.

Steeples can be modelled with pyramidal roofs. The height of the top point can be adjusted with the help of the picture. A one slope roof is created from three relevant points defining the slope plane. For this the user moves vertically one or more points of the top side of the building body.

In order to create a two, three or four slope roof, we first have to create a roof top (segment positioned thanks to its representation on various pictures). We can create a top parallel to a selected edge on the top polygon

of the building body. Otherwise, the top is arbitrarily placed in the centre of the top polygon. Then, the user interactively adjusts the roof top with the help of the background pictures and the point manipulator.

First case: the user selects the top and two points on the top polygon (one point per main slope). These points have to be at their correct place to define the corresponding slope plan. The other points on the top polygon will be vertically modified to create planar slopes. To create the sided slopes, we use the closest points to each extremity of the top roof. If other points have to be used, the second case has to be followed.

Second case: the user selects four correctly placed points on the top polygon to define the side slopes.

However, if he knows the roof slopes, the user can build the roof with these data with no help from the picture.

One can modify some points of the roof if necessary (adjustment according to other pictures), the various quadrilateral sides may not be planar any more. A command makes them planar by computing the planar shape closest to the original shape. One can notice the case of quasi-vertical polygons that can be modified to be vertical.

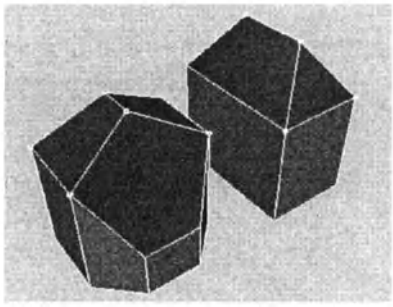


Figure 3. Examples of four slope roofs

3.2. Complex roofs

A L, U or T shaped roof is a concatenation of simple roofs according to assembling rules. In order to create a composed roof, we first have to cut the ground polygon of the building into several polygons: each part will be separately extruded and be covered with one or several simple roofs in the case of superimposing. Finally, we define the connection between two simple roofs *(Figure 4)*.

Primary and secondary roof connection: the secondary roof is extended to lay on the primary roof. The top of the secondary roof was only approximately created towards the primary roof. Several cases have to be

distinguished according to the height of the top polygons and roof tops of the two parts.

Algorithm:

1. Stretch the roof top of the secondary roof until it intersects the primary roof (not possible to continue if there is no intersection)
2. **If** the top polygon of the building body of the secondary building is lower than the top polygon of the building body of the primary building then
   Points have to be added to the secondary roof to fit the primary building (Figure 4a).
   else
   Points have to be added to the secondary body to fit the primary building (Figure 4b): the two side edges are extended until they intersect the primary roof.

Stuck roofs along a mutual wall: the two building bodies need to have a mutual wall. The two simple roofs are then stuck together along this wall (Figure 4c).

Algorithm:

1. look for the common wall
2. stretch the two roof tops to this common wall
3. modify the two roofs according to the new tops.

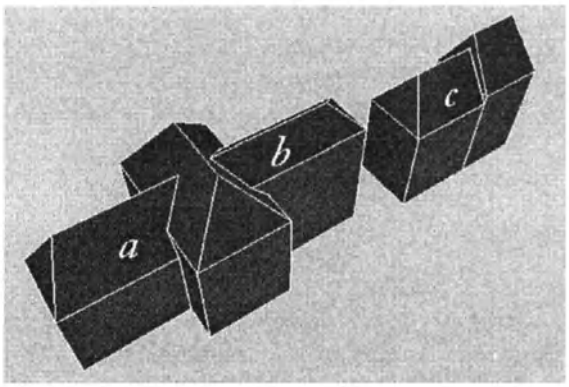


Figure 4. Composed roofs

7. RESULTS AND APPLICATIONS

Our application context was the city of Nancy in France for which some areas have been reconstructed. Pictures have been taken from the streets and from various buildings on which we were authorized to go upstairs (religious monuments, hotels, commercial centers, ...). We used our Olympus digital camera. Aerial photographs can be useful for the cutting of ground polygons and also for the horizontal placement of the roof tops. They are no helpful

for neither polygon extrusion nor for roof construction because vertical lines are not visible (except if they are taken at quite a low level). Some pictures have been used as textures for polygonal surfaces, giving more reality effect to the simulation. Pictures must be taken perpendicularly to the surface that interests us, thus avoiding perspective vision problems (balconies, window-blinds). Furthermore, pictures have been unbent, and cleaned from undesired foreground objects.

Figure 5 shows results for the area around the cathedral: (a) is the result of the viewpoint recovery, (b) is the 3D reconstruction of buildings, (c) and (d) are examples with textures.

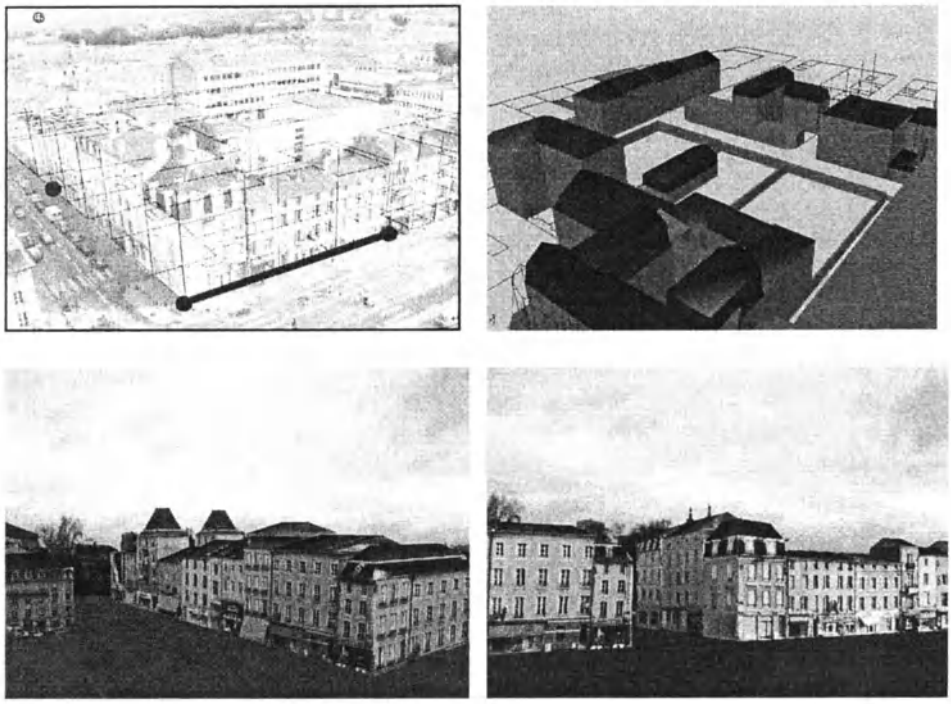


Figure 5. Results for the city of Nancy

8. FUTURE WORKS AND CONCLUSION

We have presented in this paper our system for interactive 3D rough reconstruction of urban areas from cadastral plans. However the method can be applied to any kind of known 3D data of an existing scene.

Perspectives for this work are principally to deal with curved roofs. Medina has been developed with Open Inventor library. We plan to write it

under the Maya environment. Another possibility is to develop the two modules (the automatic one and the interactive one) in another modeller environment: Maya's functionalities do not correspond to what urban designers or architects need. A solution would be to use AutoCad.

9. REFERENCES

Allani-Bouhoula, N., and J.P. Perrin, 1998, "La Reconstruction Tridimentionnelle de Tissus urbains", *First International Conference on New Technologies for Decision Making in Civil Engineering*, Montréal, p721-732.

Berger, M.O., C. Chevrier and G. Simon, 1996, "Compositing computer and video image sequences: Robust algorithms for the reconstruction of the camera parameters", *The International Journal of The Eurographics Association, Computer graphics forum*, p....

Chevrier, C, 1996a, *Génération de séquences composées d'images de synthèses et d'images vidéo*. PHD Thesis, Université Henri Poincaré, Nancy.

Chevrier, C, 1996b, "Handling interactions between real and virtual worlds", *Proceedings of the International Conference Computer Graphics International'96*, p 115-125.

Collins, R.T., A.R. Hanson, E.M. Riseman and H. Schultz, 1995, "Automatic extraction of buildings and terrain from aerial images", *Automatic Extraction of Man-Made Objects from Aerial and Space Images*, A. Gruen, O. Kuebler, P. Agouris, Birkhauser Verlag, Ascona, Switzerland, p.169-177.

Devernay, F and O. Faugeras, 1995, "Automatic Calibration and Removal of Distortion from scenes of structured environments", *Proceedings of the SPIE Conference on Investigate and Trial Image Processing*, 2567, SPIE, San Diego, CA, July 1995.

Faugeras, O, S. Laveau, L. Robert, C. Zeller and G. Csurka, 1995, " 3D Reconstruction of Urban Scenes from Sequences of images", *Automatic Extraction of Man-Made Objects from Aerial and Space Images*, A. Gruen, O. Kuebler, P. Agouris, Birkhauser Verlag, Ascona, Switzerland.

Horaud, R and O. Monga, 1993, *Vision par ordinateur, Outils fondamentaux*. Hermes, 14, rue Lantiez, 75017 Paris, fisrt Edition.

Peuchot, B, 1994, "Utilisation de détecteurs subpixels dans la modélisation d'une caméra", *Actes 9ème Congrès AFCET Reconnaissance des Formes et Intelligence Artificielle*, Paris, pages 691-695.

Simon, G and M.O. Berger, 1999, "Registration with a Zoom Lens Camera for Augmented Reality Applications", *Second International Workshop on Augmented Reality*, San Francisco.

Toscani, G, 1997, *Systèmes de calibration optique et perception du mouvement en vision.* PHD Thesis, Université de Paris-Sud, Orsay.

Wernecke, 1995, *The Inventor Mentor : programming object-oriented 3D graphics with Open Inventor*. Addison-Wesley Publishing Compagny.

Canoma, 1999 http://www.metacreations.com/products/canoma

Maya, 2000 *Using Maya Basics* and *MayaLive*, Alias WaveFront.

PhotoModeler 2000 http://www.eossystems.com

RealViz, 2000 http://www.realviz.com

Evolutionary Automata for Suburban Form Simulation

Luca Caneparo and Matteo Robiglio
Dipartimento di Progettazione Architettonica, Politecnico di Torino

Key words: Urban morphology, Model based design support system, Urban design, Landscape, Genetic programming, Cellular automata.

Abstract: The paper outlines a research project to develop a dynamic simulation of suburbanization processes. The approach to simulating suburban form relies on modelling different interacting processes on various scales. Two layered models are implemented, the Socio-Economic and Zoning model and the Suburban Form model, respectively by means of cellular automata and genetic programming. The Socio-Economic and Zoning model simulates exogenous factors and endogenous processes of large-scale suburban dynamics. The model approximates the area by means of a rectangular grid to the scale of hundred meters. The Suburban Form model uses a smaller grid, to the scale of meters, and is three-dimensional. The resulting dynamic, 3D, fine-scale model will create scenarios of suburban growth, allowing evaluation of their consequences on built environment and landscape.

1. INTRODUCTION

The paper outlines an ongoing project to develop a dynamic simulation of suburbanisation processes in Europe, through a collaborative research effort between the European Commission's Space Application Institute and the Department of Architectural Design at the Politecnico di Torino.

Urban form is the unstable result of complex interactions of site, history, economy and milieu. In suburban settings this interplay is faster and straighter than in city cores. In city cores, density in itself makes change often difficult to track, slow to accomplish, and, more important, highly influenced by individual site-specific conditions such as ownership, real

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 767–780.
© 2001 *Kluwer Academic Publishers.*

estate, memorial and artistic values, occupation etc. The "thickness" of the compact city core thus makes it difficult (and maybe useless) to simulate its change, as the model, in order to be accurate, should take in account so many non-typical aspects, that it would become something similar to Borges's one-to-one map of the world. Suburban edges growth occurs in simpler conditions, where growth factors can be reasonably typified into general categories and thus reliably modelled. Its peculiar features are low-density, mixed land use, incremental growth, fragmentation, multiplicity, and a considerable autonomy of the many, anonymous local actors (Boeri, Lanzani and Marini, 1993). Even when "rules" (plans, codes, laws) are applied – as it is in most part of Europe – the actual outcome in a built landscape is largely unforeseen, and maybe undesired. This is due to the constant interplay between rules and local actors, who freely "interpret" – actually re-write – any rule to adapt it to their own needs and wills (Boeri, 2000).

We can even portray suburbia as an "auto-organising" system (Lanzani, 1991). We mean by this that its change is neither mechanically directed from the outside as in a hierarchic system, in which the large scale determines the small scale (Holland, 1975), nor completely self-standing, with no actual interplay occurring between inside and outside. The notion of auto-organisation rests on the analogy between the city/territory and a biological organism, deeply rooted in the history of architecture (Kruft, 1985). The analogy traditionally leads to a mechanist or organic interpretation of the city/territory, according to the dominant interpretation of biological life. Cognitive sciences, stressing the importance of individuals versus the positivist dominance of species, according a central role to describing dynamic processes rather than states, relations rather than scopes, offer new keys to understanding of emerging territorial assets, by means of a deeply reviewed old analogy (Maturana and Varela, 1980).

Autonomy implied in the notion of territory as an auto-organising system is a serious challenge to urban planning and design: perhaps the most radical since planning and design have become specific disciplinary, technical and professional fields. It may be of some use to remember that our activity field is an historical construct, whose cultural roots can be traced to the XV century Italian Renaissance (Choay, 1986), but which became generalised, structured and socially recognised in the XVIII and XIX century's processes of modern State organisation (Picon, 1988), reaching its peak in '50es post-war-reconstruction and '60es welfare societies (Picon and Desportes, 1997). A constant and profound relationship (dependence) on power and state has somehow inscribed hierarchy and authority in the genetic code of architecture and urban planning, making the final user (the citizen, the inhabitant, the visitor) into a voiceless quantity, incapable of individual choice (Tosi, 1994) (Hall, 1988). When asked by Doctor Jaoul's little

daughter about her room in the father's new house, Le Corbusier answered: "Go and play, little darling, *I* know what *you* need."

The difference between Doctor Jaoul's daughter and today's actors is that the latter can do, and actually do, without Le Corbusier's advice, shaping their own built environment.

Simulating suburban change means taking a step forward and taking into account the wide range of possible results a self-organising system can produce, and may also be a step forward in regaining a role in suburban growth for architects, planners and designers.

2. WHY A MODEL, WHAT FOR A MODEL?

Let's take an example. Driving from Kent to Veneto (from London to Venice) through Flanders, Netherlands, Baden-Wüttenberg, non-alpine Switzerland and sub-alpine Brianza, we would observe the recurrence of a new urban (or suburban) form of settlement, made up of small single-family houses (cottages, fermettes, boerderettes, villette) independent or clustered, commercial facilities, storage centres, small enterprises, showrooms, car sellers, interchange parking, sport grounds, leisure parks, high-tech farming etc, all of them aligned in strips of various depth along ring roads and intercity roads (Figure 1). The mix might locally be slightly different, and quality of buildings tends to decrease going southwards, but the result is roughly the same everywhere. Largely un-planned and incremental, this kind of linear suburban growth is just one of new urban types that underline the role mobility, accessibility - and thus infrastructure - have acquired in contemporary urban networks (ITATEN, 1996) (Baart and Metz, 2000).

Figure 1. "all of them aligned in strips of various depth along ring roads and intercity roads."

2.1 The Problem

How did this all happen? Not due to intentional planning. Infrastructure, of course, was planned (and in some cases, designed). Urbanisation was planned. Zoning was planned. Facilities were foreseen. Buildings where allowed, and designed. But everything happened in separated fields, with

separated rules and knowledge. The final outcome was never taken into account as a whole, as a landscape. Neither where the role of local actors are taken into account. In fact, usually infrastructures where not meant to be urbanised. They became urbanised due to the constant pressure of a multitude of different actors and interests, any of them acting alone to get their fair share of (sub) urbanisation: families, small firms and mall chains, to mention the most frequent.

Could it have been avoided? We guess not. We otherwise cannot understand why it happened in countries with so different administrative systems, many of them really efficient in enforcing plans and rules. Could it have made into a different landscape? We guess it could have been.

What we learn from the strips' case is: first, that unexpected side- and cross-effects are more important than correctness of original choices (what happens on the sides of the highway is more crucial than the highway's section or trace); second, that far more actors than expected have been able to play a key-role, although their small-scale individual actions where not coordinated into a clear, large-scale strategy.

2.2 The Model

The capability of modelling different possible paths of suburban growth would allow planners and designers a pristine evaluation of the consequences of design and planning strategies, in their whole complexity. The model should not only allow simulation of possible future states - by simulating the multiplicity and diversity of local actors' reactions to design and planning choices - but also recursive interaction with policy, planning and design making by means either of successive corrections and adaptations, or of radical changes of strategy if needed (choice → simulation → evaluation → adaptation/change → simulation ...). This means leaving the current "curative" approach of "damage containment" for a more effective preventive integrated design and planning strategy.

Modelling suburban phenomena is unfeasible, at least with the mechanicistic or reductionist apparatus. The magnitude of the phenomena, the interplay of the causes and effects, and the role of local actors in the comprehensive process is such that it requires new intellectual and experimental instrumentation.

We need an holistic approach to model suburban dynamics, considering the interplay between individual behaviours of local actors (e.g. the decision to settle, the choice of a location, the seek for visibility or, on the contrary, privacy, the choice of a building typology and construction system, the necessity for adaptation and further change when facing new needs, etc.) and global processes (e.g. general planning, socio-economic trends, market

dynamics, infrastructure, mobility and accessibility etc.), up to regional scale and vice versa. Therefore major interventions are applied from the top-down, for instance planning and zoning actions, major infrastructure or facilities interventions, as in fact it still.

Furthermore, the model has to consider three-dimensional phenomena. Contemporary suburbanisation processes have clearly shown the limits of a bi-dimensional attitude, reducing planning and urban design to the mere outlay of zoning areas. But 3D modelling would also allow a more direct interplay between planners and designers on one side, decision-makers, citizens and users on the other. Their visual evidence would subtract evaluation and decision from the elite hands of technicians, making by means of a communicative language an open public discussion and participation possible. This should result in a further increase of efficacy, as public debate should no longer get stuck in positional conflict about "what" (are we for or against the new highway?) but evolve into open negotiation about "what/how" (what happens if the highway is built? and if not? who is damaged? are there possible compensations? can intersections be changed? can design be improved? ...) (Bobbio, 1999). Far from disparaging the technician's competence, such a process would enhance its key role in fostering new and creative solutions, offering to public debate the wider possible range of assets.

Our approach to simulating suburban form relies on modelling different interacting process at various scales. In the view of the validation with real European scenarios, we are implementing two layered models, the Socio-Economic and Zoning model and the Suburban Form model, respectively by means of cellular automata and genetic programming.

3. SOCIO-ECONOMIC AND ZONING MODEL

The Socio-Economic and Zoning Model (SEZ) simulates exogenous factors and endogenous processes of large-scale suburban dynamics. The model approximates the area by means of a rectangular grid at the scale of hundred meters, i.e. square cells of 100 m side.

The space of the model is anisotropy: at each cell is associated a vector representing respectively actual use, location, accessibility and zoning status of the area, making it more or less suitable for development.

The dynamics is simulated by means of Cellular Automata: the state of any cell is related to the states of the neighbouring ones.

3.1 Background on Cellular Automata

Cellular automata were originally introduced by von Neumann (1963 and 1966) and Ulam (1974) as a possible model of biological systems.

For the purposes of the SEZ model, a cellular automaton is a discrete dynamical system. Space, time, and the states of the system are discrete. Each site –called a cell– in a regular spatial lattice or array –the grid– can have any one of a finite number of states. The states of the cells in the lattice are updated according to the rules. That is, the state of a cell, at a given time, depends only on its own state one time step previously, and the states of its nearby neighbours at the previous time step. All cells on the lattice are updated synchronously. Thus the state of the entire lattice advances in discrete time steps (Batty and Xie, 1994) (Cecchini, 1999) (Couclelis, 1997) (Engelen, White and Uljee, 1997) (Wagner, 1997) (Engelen,. Geertman, et al., 1999).

The mathematician's definition (Wolfram, 1994) of a cellular automaton is: a regular lattice of sites, where each site takes on k possible values, and is updated in discrete time steps according to a rule that depends on the value of sites r in some neighbourhood around it.

Conventionally, d=dimension, k=states per site, r=radius. In the SEZ model are assumed d=2, k=24 and r=8.

A d-dimensional cellular automaton takes as its underlying space the lattice S^Z (Z=integers, infinite in both positive and negative directions) where S is a finite set of k elements. The dynamics are determined by a global function

$F : S^Z \rightarrow S^Z$

whose dynamics are determined "locally". A "local (or neighbourhood) function" f is defined on a finite region

$f : S^{2r+1} \rightarrow S^Z$

The global function F arises from f by defining

$F(c)_i = f(c_{i-r}, \ldots, c_{i+r})$

3.2 Applying Cellular Automata to Socio-Economic and Zoning Model

In essence, the application of cellular automata to the SEZ model determines the status of an individual cell according to the states of the cells within its neighbourhood. The neighbourhood consequence is evaluated for each of the states per site, which the cell could be converted to. At run time the cellular automaton generates attraction or repulsion ("forces" dynamics) for each cell according to the associated status and transition rules. Since the dimension of the cell is a hectare, in the SEZ model the neighbourhood

radius is 0.8 km. This size of the neighbourhood properly simulates micro-scale suburban dynamics, while larger or macro-scales can be simulated by means of coarser granularity of the model, i.e. larger cells. That is why, it is considering layering various cellular automata models of the same urban area, but at various scales and modelling different phenomena.

The model is developed by the European Commission's Space Application Institute in collaboration with RIKS Institute, and is validated with historical data series over the last 40 years for several European cities (Figure 2).

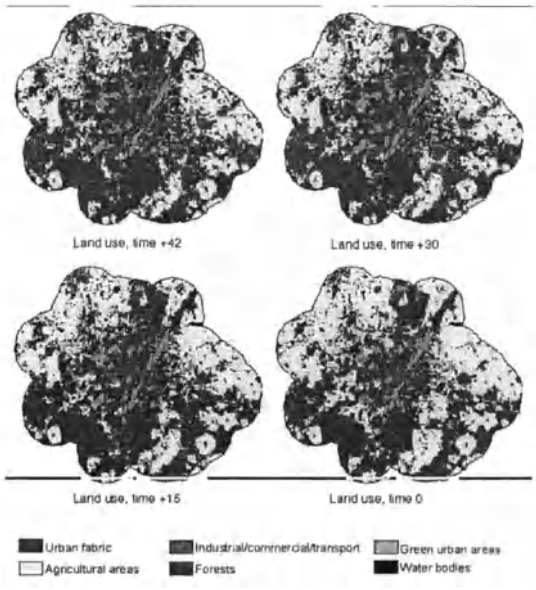


Figure 2. Munich, simulated land use over the last 42 years.

4. SUBURBAN FORM MODEL

The Suburban Form model (SF) uses a thinner grid than the SEZ model, at the scale of meters, and a three-dimensional grid instead of a bi-dimensional one. At each cube of the grid is associated its state (e.g. residential, industrial, commerce, tertiary, services, agricultural…). The 3D grid cell or cube is also referred to as a *voxel* (volumetric elements) (Jones, 1989).

The push to change, from the initial state, is an input from the SEZ model, measured as a delta in density for each land use: greater the demand-density of a certain destination for a specific area, greater is the "pressure" to change the existing status of the area (Figure 3).

The model pursues the maximisation of use and economic values for local actors.

A set of transition rules defines the constraints the model has to fulfil. They are global, apply to the entire model, or specific, domain dependent. These rules are:

- Building and zoning regulations (i.e. maximum heights, minimum distances etc.);
- Accessibility (e.g. secondary or private roads, entrance/s),
- Observed usual behaviours of local actors (e.g. search for visibility, use of simplified construction grids etc.);
- Typological and constructive recurrences (e.g. residential and industrial types, prefab building spans etc.);
- Specific criteria apply when the input from the SEZ model overcomes the capability of "greenfield" development by converting agricultural/unused areas and/or expanding existing buildings. These criteria shift from incremental (plot-by-plot) to radical change of whole areas, trying, for instance to aggregate fragmented plots or to intensify building by adopting more easily rented types. The greater the requirements from the SEZ model, stronger the pressure to change the present use and form of the area.

The matching of the global and specific criteria is an NP-hard optimisation problem (Garey and Johnson, 1979): finding the exact solution of the problem for a real urban scenario is prevented by exponential explosion. The SF model implements genetic programming to discover and learn best procedures to pursue the optimisation functions (Broughton, Tan, and Coates, 1997) (Jagielski and Gero, 1997).

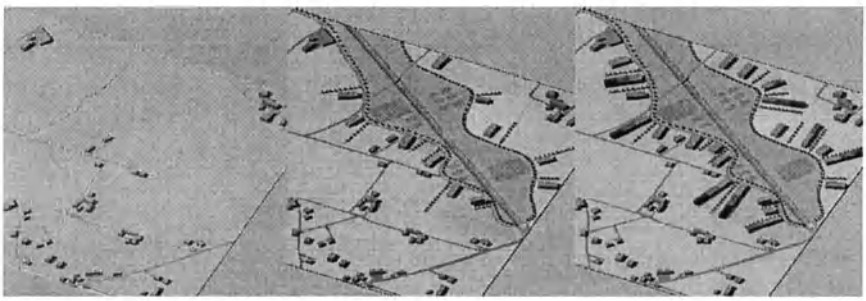


Figure 3. Suburban form simulation at time 0 (left), +5 (centre) and +15 (right).

4.1 Background on Genetic Programming

Genetic Programming (GP) is a biologically inspired methodology that automatically generates a computer program from a high-level statement of

a problem's requirements. It was developed by John Koza (1992, 1994 and 1999), starting from the genetic algorithm described in Holland's book *Adaptation in Natural and Artificial Systems* (1975). Koza suggests that the desired program should evolve itself from the initial statement/s through the evolution process. Instead of writing the program to fulfil the requirements and pursue the goals of the SF model, GP methodologies search the space of possible computer programs for best ones, according to the defined goals. A population of executable computer programs is created, satisfying the given requirements. The solutions provided by the individual programs are evaluated, one against the others: worst programs are abandoned, while the best ones further "reproduce" themselves according to genetic laws, i.e. sexual recombination (crossover), mutation, gene duplication or gene deletion. The reproduction and evaluation process is repeated over and over, until the results, the goals of the model are satisfactory approximated.

General GP methodology can be summarised in the following three steps (Koza, 1994):

1. Randomly create an initial population of individual computer programs.
2. Iteratively perform the following substeps on the population of strings until the termination criterion has been satisfied:
   Assign a fitness value to each individual in the population using the fitness measure.
   Create a new population of strings by applying the following three genetic operations. The genetic operations are applied to individual string/s in the population selected with a probability based on fitness (with reselection allowed).
      Reproduce an existing individual string by copying it into the new population.
      Create two new strings from two existing strings by genetically recombining substrings using the crossover operation at a randomly chosen crossover point.
      Create a new string from an existing string by randomly mutating the character at one randomly chosen position in the string.
3. Designate the string that is identified by the method of result designation (e.g., the best-so-far individual) as the result of the genetic algorithm for the run. This result may represent a solution (or an approximate solution) to the problem.

4.2 Applying Genetic Programming to Suburban Form Model

The comprehensive suburbanisation processes are simulated disaggregating them into subproblems, functions in the GP terminology, each considering an individual urbanisation objective or task. These functions are divided into four categories: plot aggregate, road, building, and parking.

The functions are competitively applied to the initial area in a concurrent process. Namely the functions in the SF program concurrently work on the area, and eventually its adjacent plots, until the building and its facilities are completed. The suburbanisation process, generated by each individual computer program, is evaluated to determine the fittest. Until the goals of the model are satisfactory approximated, the evolution process is iterated and a new generation of computer programs is created by means of reproduction, crossover and mutation. Since a constrained syntactic structure is involved, crossover is performed so as to preserve this syntactic structure in all offspring.

4.2.1 Initial Area

An initial area is the given input to the GP program. This area has a shape, from cadastral maps, a possible new destination and an incremental density (from the SEZ model).

4.2.2 Plots Aggregate

Of the initial area it is evaluated whether the new destination and density are compatible with (1) eventually existing buildings (2) the area extent. If not, the "plot aggregate" function is applied trying to enlarge the area aggregating adjacent plots, minimising the required financial investment.

4.2.3 Road

Accessibility to the area is considered in relation to the (new) destination and density. The "Road" function tries to maximise the accessibility creating new roads, minimising the economic cost, i.e. the land occupation vs. minimum path to building entrance/s and eventual facility area/s.

4.2.4 Building

The "Building" function inserts a new building in the area. This function is the root of a subtree of functions maximising respectively (1) building and zoning regulations (i.e. maximum heights, minimum distances etc.), (2) usual behaviours of local actors (e.g. search for visibility, use of simplified construction grids etc.), (3) typological and constructive recurrences (e.g. residential, tertiary or industrial types, prefab building spans etc.).

4.3 Domain Knowledge and Genetic Programming

GP offers techniques to search the space of computer programs for the fittest, but this search is "blind". Thus it is advantageous to incorporate some knowledge to reduce or lead the search. Because the space of possible computer programs which GP methodologies explore is huge, the search can take too long to pursue a satisfactory solution. Furthermore GP can miss promising solutions, which design practice teaches us are effective.

We are working to incorporate two kinds of knowledge, respectively on the suburbanisation processes, and on spaces and objects.

The knowledge on the process can be represented into the formalisation of the problem, which GP has to search, in variation operators known as useful, or into the performance index in the form of known behaviours, for instance behaviour of local actors or typological and constructive recurrences. Incorporating such knowledge focuses the genetic search, producing a more effective exploration of the space of computer programs.

The knowledge on spaces and objects is given directly by the architect or planner, whose experience can suggest promising solutions. These solutions are often the result of a creative synthesis, but often without awareness of the underlying creative process. To incorporate this knowledge in the SF model, we are pursuing direct designer's intervention during the GP process. The designer acquires more direct control of the GP process through:

- display of the solutions provided by the individuals programs of the current generation,
- pause of the genetic process,
- modification of a design solution/s or the creation of a new one,
- continuation of the GP process with a new generation considering the modification manually introduced.

To implement these four steps, the system should integrate:

- visual clustering of the individuals, as they could sum to hundreds of thousands for each generation,
- an editor to interact with the form and its attributes,

– a map the changes from the editor inside the internal GP representation, which is an executable program.

Our present approach is associating the user's commands in the editor directly to executable statements, which the GP can immediately reuse.

5. CONCLUSIONS

To model suburban processes we are facing some core problems posed to modern science: the magnitude of the phenomena, the interplay of causes and effects and the role played by local actors and phenomena in the comprehensive process. The title *"evolutionary automata"* states our exploration in instrumentations for simulating the peculiar processes of production of space in a self-organising territory, by means of cellular automata and genetic programming. This integration is still undergoing.

The SEZ model is in the calibration phase, and is proving good results for several European cities. The analysis of the validation and calibration processes is behind the scope of the present paper (cf. White, Engelen, et al., 1999).

The SF model is still in an "accumulative" phase, where knowledge-rules on typological and constructive recurrences (e.g. residential, industrial types, commercial etc.) are added to the model. The model still has to be put through the experimental cycle: run the model, analyse results, compare them with reality... Genetic programming is proving its capability of synthesise-design suburban environments, entailing the site, accessibility and building, considering topology, dimension and layout issues. For prototypal applications, where it is relevant the capability to quickly change and experiment with the rules and parameters of the model, GP is demonstrating demanding for both the users, because translating high-level rules into executable structures requires a major conceptualisation effort, and for the computers, because GP search is very CPU demanding for the creation of the population and the generations.

On the other hand, it is remarkable the capability of GP to generate innovative solutions and to discover regularities in the space problem. We expect experimentation with a greater number of cases to shed light on rules underling the evolution of suburbanisation dynamics with validity transcending the specific cases.

If GP will demonstrate, through the calibration and validation phases, an acceptably accurate instrument for 3D dynamic and interactive simulation, it would offer an effective tool for decision-making and design improvement at an intermediate scale, between the region and the single plot. The scale where built landscapes are shaped, planning decisions get physical and have

to interact with local actors, environment becomes a specific site. The scale where the current crises of planning and urban design facing multiplicity and autonomy is taking place.

6. REFERENCES

Baart, T., and T. Metz, 2000, *Atlas van de verandering*, NAI Uitgevers, Rotterdam.

Batty, M. and Y. Xie, 1994, "From cells to cities", *Environment and planning B*, 21, pp. 31-48.

Bobbio, L., *La democrazia non abita a Gordio*, Franco Angeli, Milano.

Boeri, S., 2000, *USE – Uncertain States of Europe*, in: *Mutations*, Centre Arc en Réve, Bordeaux.

Boeri, S., A. Lanzani and E. Marini, 1993, *Il territorio che cambia*, Abitare Segesta, Milano.

Broughton, T., A. Tan, and P.S. Coates, 1997, "The Use of Genetic Programming in Exploring 3D Design Worlds", in: Junge, R. (ed.), *Conference Proceedings of CAAD Futures 1997*, München, pp. 885-915.

Cecchini, A., 1999, *Meglio meno, ma meglio automi cellulari e analisi territoriale*, FrancoAngeli, Milano.

Choay, F., 1986, *La regola e il modello*, Officina, Roma.

Couclelis, H., 1997, "From cellular automata to urban models: new principles for model development and implementation", *Environment and Planning B*, 24, pp. 165-174.

Engelen, G., R. White and I. Uljee, 1997, "Integrating constrained cellular automata models, GIS and decision support tools for urban planning and policy-making", in: Timmermans H. (ed.), *Decision Support Systems in Urban Planning*, E&FN Spon, London, pp. 125-155.

Engelen, G., S. Geertman, P. Smits and C. Wessels, 1999, "Integration of GIS and Dynamic Modelling for Regional Planning, in: Geertman, S., S. Openshaw and J. Stillwell (eds.), *Geographical information and planning*, Springer, Berlin.

Garey, M. and D. Johnson, 1979, *Computers and Intractability*, W.H. Freeman, San Francisco.

Hall, P., 1988, *Cities of tomorrow*, Basil Blackwell, Oxford.

Holland, J., 1975, *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

ITATEN, 1996, "Indagine sulla trasformazione del territorio italiano", in: A. Clementi, G. Dematteis and P.C Palermo (eds.), *Le forme del territorio italiano*, Laterza, Roma-Bari.

Jagielski, R. and J.S. Gero, 1997, "A Genetic Programming Approach to the Space Layout Planning Problem", in: Junge, R. (ed.), *Conference Proceedings of CAAD Futures 1997*, München, pp. 865-884.

Jones, C.B., 1989, "Data Structures for Three-Dimensional Spatial Information Systems in Geology", in: *International Journal of Geographical Information Systems*, 3, 1, pp. 15-31.

Koza, J., 1992, *Genetic programming: on the programming of computers by means of natural selection*, MIT Press, Cambridge (Ma).

Koza, J., 1994, *Genetic programming II: automatic discovery of reusable programs*, MIT Press, Cambridge (Ma).

Koza, J., 1999, *Genetic programming III: darwinian invention and problem solving*, Morgan Kaufmann, San Francisco.

Kruft, W. A., 1985, *Geschichte der Architekturtheorie*, Beck, München.

Lanzani, A., 1991, *Il territorio al plurale*, Franco Angeli, Milano.

Maturana, H. R., and F. J. Varela, 1980, *Autopoiesis and cognition. The realization of the living*, Reidel, Dordrecht.

Picon, A., 1988, *Architectes et engenieurs au siècle des lumières*, Presses de l'École National des Ponts et Chaussées, Paris.

Picon, A., and M. Desportes, 1997, *De l'espace au territoire*, Presses de l'École National des Ponts et Chaussées, Paris.

Tosi, A., 1994, *Abitanti*, il Mulino, Bologna.

Ulam, S., 1974, "Some ideas and prospects in biomathematics", *Ann. Rev. Bio.*, 255.

von Neumann, J. 1963, "The general and logical theory of automata", in: von Neumann, J. *Collected Works, Design of computers, theory of automata and numerical analysis*, Taub, A. H. (ed.), Pergamon, Oxford.

von Neumann, J. 1966, *Theory of Self-Reproducing Automata*, Burks, A. W. (ed.), University of Illinois Press, Urbana.

Wagner, D., 1997, "Cellular automata and geographic information systems", *Environment and Planning B*, Vol 24, pp. 219-234.

White, R., G. Engelen, I. Uljee, C. Lavalle and D. Erlich, 1999, Developing an Urban Land Use Simulator for European Cities, JRC Technical Paper, Ispra.

Wolfram, S., 1994, *Cellular Automata and Complexity*, Addison-Wesley Publ., Reading (Ma).

Web-Based Virtual-Reality for Collaboration on Urban Visual Environment Assessment

Zongyu Zhang, Jin-yeu Tsou, and Theodore W. Hall
The Chinese University of Hong Kong

Key words: geographic information systems, internet, urban landscape, visual assessment, virtual reality

Abstract: This research aims to facilitate public participation in urban landscape visual assessment (ULVA). To support virtual collaboration in ULVA, it is desirable to provide both quantitative analysis and 3D simulation over the Internet. Although the rendering of urban models in common web browser plug-ins often lacks vividness compared with native workstation applications, the integration of VRML modeling and Java programming proves effective in sharing and rendering urban scenes through a familiar web interface. The ULVA simulation supports not only static scene rendering, but also interactive functional simulations. They include the viewpoint setting up, view corridor and panorama generation. Although popular VRML viewers such as CosmoPlayer provide similar functions, users are often disoriented by the interface. The obfuscation inhibits people's immersion in the virtual urban environment and makes the assessment inconvenient. To eliminate such disorientation and improve users' feelings of immersion, we integrate both a two-dimensional map and a three-dimensional model of the urban area in the user interface. The interaction between 2D map and 3D world includes the matching of avatar positions, visualization of avatar posture, and the setting up of viewpoints and view corridors. To support a web-based urban planning process, the system adopts client/server architecture. The city map is managed by a specific database management system (DBMS) on the server side. Users may retrieve information for various "what if" simulations. The system automatically remodels the virtual environment to respond to users' requests.

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 781–794.
© 2001 *Kluwer Academic Publishers.*

1. INTRODUCTION

1.1 Visual Impact Assessment

Various researchers have focused on the analysis of people's perceptions of and preferences for natural landscapes. However, a systematic approach to visual impact assessment has been lacking (Wherrett, 1996b). Visual impacts arise from changes in land use and management, development of buildings and structures, and less commonly, changes in production processes and emissions. In the life of a built project, many different sources of impact occur at various stages, including construction, operation, decommissioning, and restoration (IEATLI, 1995). In fast developing areas such as China, the intermediate effects are often ignored by planners, while the final visual impact warrants public attention. Visual impact assessment is mainly concerned with: (a) the direct impacts of development upon views of the landscape through intrusion or obstruction; (b) the reactions of viewers who may be affected; and (c) the overall impact on visual amenity, which is often a cumulative effect of many minor processes and can range from degradation to enhancement (IEATLI, 1995).

1.2 Public Participation

As a planning process concerned with public perceptions, urban landscape visual assessment (ULVA) wants public participation. For natural resource managers to plan for a healthier environment, and to elicit public and political support for such plans, two needs have been identified: (1) to predict the responses of various public groups to changes in the environment, for some of whom the visual impact may be a dominant factor, and to plan to minimize any negative impact; (2) to communicate the effects of proposed changes to other agencies and public review groups to facilitate decision making (Orland, 1994). Two fundamentally different approaches to evaluation can be distinguished: one, a professional based approach, in which the evaluation is carried out by an expert or a group of experts; the other, a publicly based approach, in which the evaluation is carried out by a number of lay people representing the public or different social groups (Lange, 1994). The empowerment of public participation in the planning process is often diagrammed as a ladder. The degrees of public involvement in environmental decision-making ascend through: the public's basic right to know; actively informing the public; public participation in defining interests, actors, and agenda; public participation in assessing risks and recommending solutions; public participation in final decision-making (Kingston, 1998).

Traditionally, simulation has played several important roles in analyzing visual impact. In the context of landscape aesthetic policy, simulation can serve: (1) the communication to the public of the probable consequences of environmental modifications; (2) the definition of perceptually based performance standards for land use regulation; (3) the assessment of monetary penalties for aesthetic damages. As useful tools for such impact assessment, a wide array of landscape simulation methods have been developed and implemented. These include: plans, diagrams, elevations, perspective sketches, renderings, modified photographs (photo renderings and photomontages), slide projections, scale models, movies, videotapes, and computer graphics. Among these, computer-based methods have particularly developed in recent years and have shown great potential as simulation tools. Computer-based simulations include: two dimensional drafting and painting; three-dimensional wire frame, surface modeling, and solid modeling; image processing; and animation techniques (Oh, 1993). Briefly, all of these simulations can be categorized according to whether they are based on 2D images or 3D models. The former excels in pictorial realism, while the latter empowers the user to interact with objects in space. Recently developed virtual-reality (VR) technology can be considered as the ultimate 3D simulation for urban landscape. It allows the user to navigate in the virtual environment and to communicate with others as avatars when they share the same virtual community. Virtual reality is attracting increasing attention from planners as they try to improve public involvement in assessing the visual impact of development. Compared with traditional methods of public participation, such as questionnaires, surveys, or community conferences, the virtual environment provides individuals more freedom to explore and empowers them to discover the root of some negative impacts.

As the number of households with Internet access increases and the demographic profile of Internet users diversifies, the potential for using the World Wide Web (WWW) for public participation in planning increases exponentially (Kodmany, 2000). Now, more and more researchers are exploring the intersection of the Internet and planning, particularly using GIS on the Internet. R. Kingston outlined research that examined the potential of the Web as a means of increasing public participation in environmental decision making (Kingston, 1998). He considered traditional methods of public participation and argued that new Internet-based technologies have the potential to widen participation in the planning system. There are some clear advantages for survey participants, including privacy, freedom from time pressures and constraints, and freedom from

influence by an interviewer. All these contribute to alleviate confrontation between participants and planning agencies.

1.3 Qualified Public Participation in Visual Impact Assessment

Although public involvement in the planning process contributes to democracy and promotes the public's living standard, it is still considered unreasonable to substitute the public's myriad opinions in place of planners' professional judgements when it comes to urban landscape planning. Professionals in the field of design and environmental planning are seen to have a more sensitive appreciation of landscape quality and are also thought to be able to articulate their feelings more expressively (Dearden, 1981). Citizen interest is thought by some to be lacking in landscape evaluations because of the inherently subjective and somewhat intangible nature of the problem (Wherrett, 1996a). There is also concern about creating a critical mass of users to sustain meaningful Internet interactions. Some initial attempts at creating public discussion sites have withered because of the lack of message activity (Kodmany, 2000).

The goal of this research is to provide a collaboration mechanism between planners and the public for visual impact assessment of urban landscape. It focuses on the representation of view impact, communication between the two sides, media for communication, and the distribution and limits of authority. We have developed a web-based virtual-reality system to empower public participation in visual impact assessment of urban landscape. Common practice with traditional methods of public participation quite often involves an atmosphere of confrontation. In contrast, this system aims to empower people more freely to acquire information on site planning, express their opinions, and even object to proposals in virtual communities. However, the system may still present a barrier to public participation in visual impact assessment, due to the unfamiliarity of the public with the virtual environment. This challenges the virtual-reality system not only in its technical capabilities, such as the vividness of simulation, user interactions, and communications, but also in the design of its user interface and work style. This last item should also keep accord with the system's communicator role between the public and planners.

2. SPECIFICATIONS FOR THE WEB-BASED SYSTEM

Although the rendering of urban models in common web browser plug-ins often lacks vividness compared with native workstation applications, VRML modeling still proves effective for sharing and rendering urban scenes through a familiar web interface. Three-D simulation for visual impact assessment requires not only static scene rendering, but also various specific interactions. For example, the system should allow users to "walk" in the virtual urban landscape, select viewpoints, set up parameters such as height, range of view, and orientation, and automatically generate the scene. It should also provide panorama and view corridor simulations. In the latter, the user predefines a tour route through the virtual city, and the system generates an animation of a fly-through or walk-through along the route. Although popular VRML viewers such as CosmoPlayer provide some wander simulations, users are often disoriented when they try to "walk" through the model. The obfuscation of the user interface may distort the perception of the virtual urban landscape, interfere with people's assessment of its beauty, and make subjective assessment inconvenient. To mitigate such disorientation, we integrate both a two-dimensional map and a three-dimensional model of the urban area in the user interface. The interaction between 2D map and 3D world includes the matching of avatar positions, visualization of avatar view orientation and range, and the setting up of viewpoints and view corridors. To support web-based assessment, the system adopts a client/server architecture. The city model and map are managed by a specific database management system (DBMS) on the server side.

The 2D map is actually a web-enabled Geographic Information System (GIS) based on a client/server architecture. In our system, it not only aids the orientation of avatars for wandering in the virtual environment, but is also customized to accomplish some quantitative analysis based on intervisibility, such as the "view shed" area from given viewpoints, visual sensitivity, and cumulative visual impact exposure. The integration of web-enabled GIS and a web-based virtual environment brings new capabilities for public participation in visual landscape assessment.

Normally, in the domain of landscape evaluation, the descriptive inventory approach contains several assumptions. One is that the value of a landscape can be explained in terms of the values of its components. Another is that scenic beauty is embedded in the landscape components, that it is a physical attribute of the landscape.

However, scenic beauty depends on the observer as well as that which is observed (Arthur et al., 1977). Public perceptions may be superficial and more impulsive than a professional's. Lacking in professional training and often limited in their analytical abilities, lay people cannot articulate the basic reasons for their positive or negative evaluations of urban landscape development. As to the urban landscape planning process, they have difficulty formulating constructive suggestions for planners. It's easy to let people evaluate visual impact as good or bad. When asked to explain their evaluations, their reasoning is often unclear or faulty.

Therefore, the quality of public assessment should be checked. In our system, we provide a log mechanism for public participation to address this issue. Participants' operations and evaluations are logged into a specific database that includes their intervisibility analysis, communication, and navigation in the virtual environment. Professionals can subsequently rebuild the scene according to a participant's parameter set, which includes position, view parameters, tour routes, and other similar information. After reviewing a participant's visual experience in the virtual environment, professionals can evaluate the quality of the participant's assessment.

3. IMPLEMENTATION

Besides providing the visual analysis functions mentioned in the previous section, the system design also focuses on the interactions between the virtual environment and the GIS. We utilize the GIS as a guide map for the user when navigating through the virtual environment. This aims to reduce the negative impact on the visual assessment due to disorientation.

3.1 Modeling the World

Currently, there are two common approaches to modeling web-based virtual environments. One is to use VRML (the Virtual Reality Modeling Language) with Java programming; the other is to use the Java3D application program interface (API).

VRML has become an industry standard for graphical description of virtual worlds on the Internet. Many 3D modeling and simulation programs have the ability to export their models in VRML format. With a VRML browser, one can control the view so that any part of the model can be examined at any orientation and scale.

VRML uses a hierarchical scene graph to describe the 3D model. Entities in the scene graph are called nodes. Nodes store their data in fields. Nodes can contain other nodes and may be contained by more than one

node. By organizing the nodes and specifying their values, one can encode the geometry and topology of geographical information, and its illumination and material appearance as well. In addition, texture images, which can be obtained from digitized photographs of landscape elements, can be transformed and mapped onto the geographical surfaces. Another feature of VRML is multiple representation characterized by level of detail (LOD). Entities in the virtual world can be defined in the scene graph at various levels of detail, from coarse to fine. The VRML browser can then automatically choose and display the appropriate definition of an entity based on its distance from the viewer and its projected image size.

The External Authoring Interface (EAI) links the VRML scene graph to Java programming. This has three benefits. First, with EAI, we can take advantage of existing VRML browser features, particularly the navigation functions. Second, the VRML scene graph can be dynamically built and updated via the EAI, based on data received by Java applets that interact with the two-dimensional map of the virtual world; conversely, the applet's data can also be dynamically updated through the VRML interface. Third, a web-based system utilizing EAI and VRML can be easily accessed from any platform with a Java-enabled web browser and a VRML browser plug-in. Support for low-cost, platform-independent clients is one of the main features of utilizing VRML and Java for developing virtual environments [Kyong-Ho Kim]. There have been many success stories in using VRML with Java to develop web-based virtual-reality systems.

The Java3D API is another interface for Java applications to display and interact with three-dimensional graphics. As an API, rather than a modeling specification, Java3D provides a very flexible means to develop a real-time web-based 3D graphics system. At the same time, it requires the developer to understand the structure of any imported objects and to apply sophisticated graphic programming skills. Java3D does not specify or support any particular file format, making it tedious to create, share, and archive 3D models. To develop a practical web-based virtual-reality system solely through the Java3D API is a great challenge to developers.

For quick prototype development, it seems more practical to utilize VRML and EAI with Java (rather than Java3D) to create and manage the interaction in our virtual environment.

It is also desirable to integrate both the design and evaluation modules into one system. Unfortunately, popular GIS applications are not optimized for design and inhibit those accustomed to designing with CAD applications. Importing data from other kinds of software greatly challenges the interoperability of GIS. Although many popular GIS packages, such as Arc/Info, Arcview, or MapCAD, have specific extensions for the import of

CAD data, the semantics of data structures defined in GIS and CAD are different. Thus it is important for the model developer to customize data imported from other software to conform to GIS semantics. The reverse situation is similar. The outcomes of visual impact assessments are often stored in the GIS database as thematic maps. These need to be transformed for export to CAD or other software data formats according to their semantics.

3.2 Linking the GIS with the Virtual Environment

Because until now there is no common database management system for VRML models, we developed a specialized system to manage the inventory of entities in the virtual environment. Each entity is identified by a unique code that serves as a primary key in the database. The two-dimensional map of the urban area is managed by the GIS, in which spatial objects are classified and uniquely identified. Using these unique identifiers, the system maintains the correspondence of spatial objects between the GIS and the virtual environment. We used a customized Internet map server for the web-based GIS. We also developed a web interface for intervisibility analysis.

The interaction between the two-dimensional map and the three-dimensional virtual environmental on the client side is supported by a specific module programmed in Java. It handles the coordinate transformations and communications between the 2D and 3D representations via the EAI.

The virtual environment is created with popular 3D modeling software, such as AutoCAD, 3DStudioViz, or GIS 3D extensions. These all have the ability to export 3D models in VRML format. The export process often hides the coordinate transformations. To unveil the hidden relationship between the GIS and VRML coordinates, we first apply statistical analysis to some sample points from the two systems. The assumption is that all transformations are combinations of rotation, scaling, and translation. The relationship must linear. It can be expressed as:

$$\begin{pmatrix} X_{GIS} \\ Y_{GIS} \\ Z_{GIS} \end{pmatrix} = \begin{pmatrix} A_1 & B_1 & C_1 & D_1 \\ A_2 & B_2 & C_2 & D_2 \\ A_3 & B_3 & C_3 & D_3 \end{pmatrix} \times \begin{pmatrix} X_{VRML} \\ Y_{VRML} \\ Z_{VRML} \\ 1 \end{pmatrix}$$

The statistical results are encouraging. The regression confirms that all transformations are linear. The R and R-squared values are all nearly 1.0.

This helped us to easily deduce the relationship between the GIS and VRML coordinate systems.

Figures 1 through 5 illustrate the client-side user interface, with the GIS frame on the left and the VRML frame on the right. In Figure 4, the user applies an intersection operation between the visible area and the building theme in the GIS to determine what buildings can be seen from the given viewpoint. In Figure 5 the user provides feedback on his visual impact assessment. The feedback panel offers a choice of three viewing operations as the basis of the assessment: view from a predefined viewpoint; fly-through along a predefined tour route; or view as panorama. In the illustrated case, the user judged the development to have a negative impact on the original scene as viewed from the selected point. He stated the reason for his assessment: "the building destroys the skyline of the natural scene." Figure 6 diagrams the system's client/server architecture.

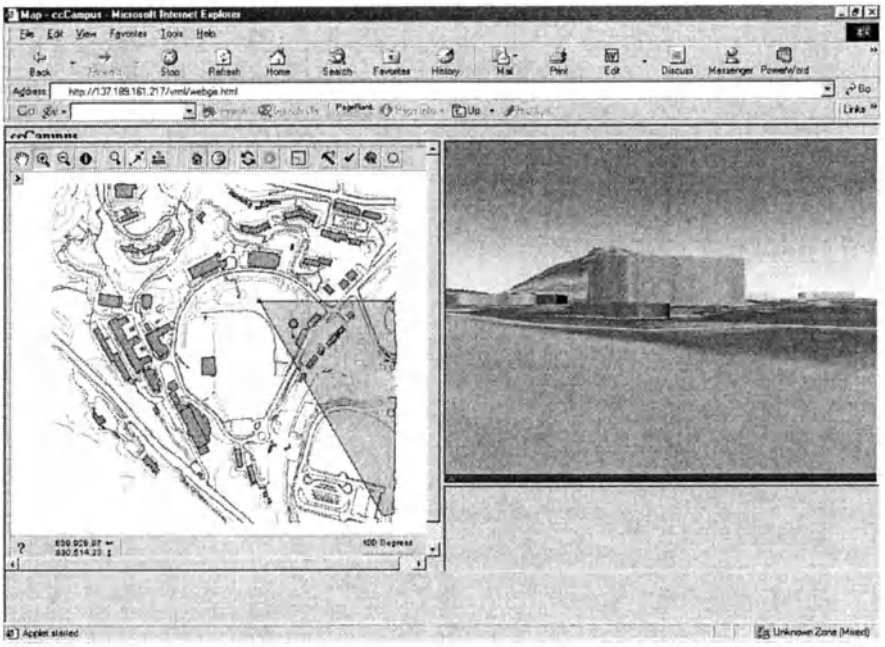


Figure 1. Transforming the view area from the VRML environment onto the GIS map.

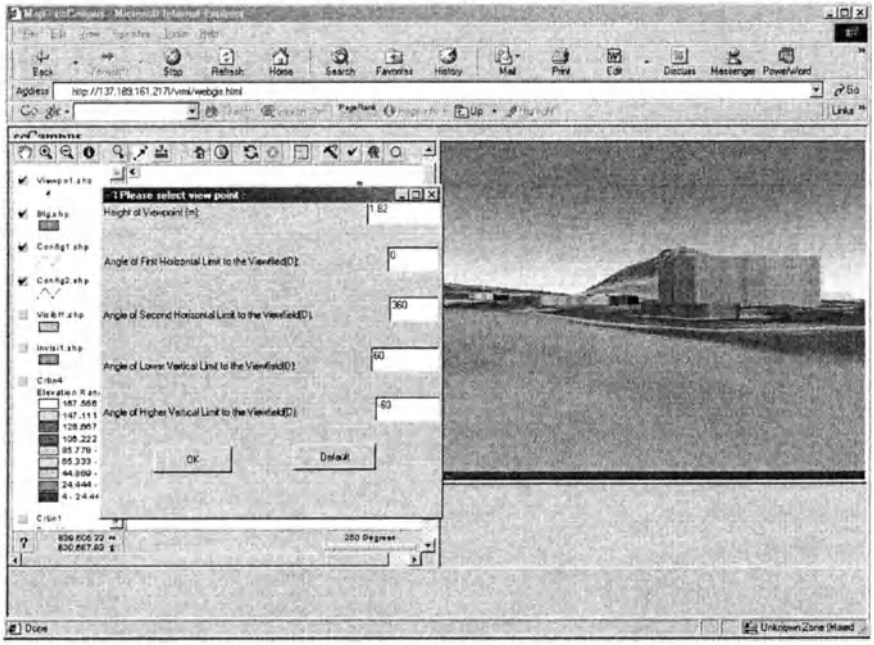


Figure 2. Panel for viewport parameter specification.

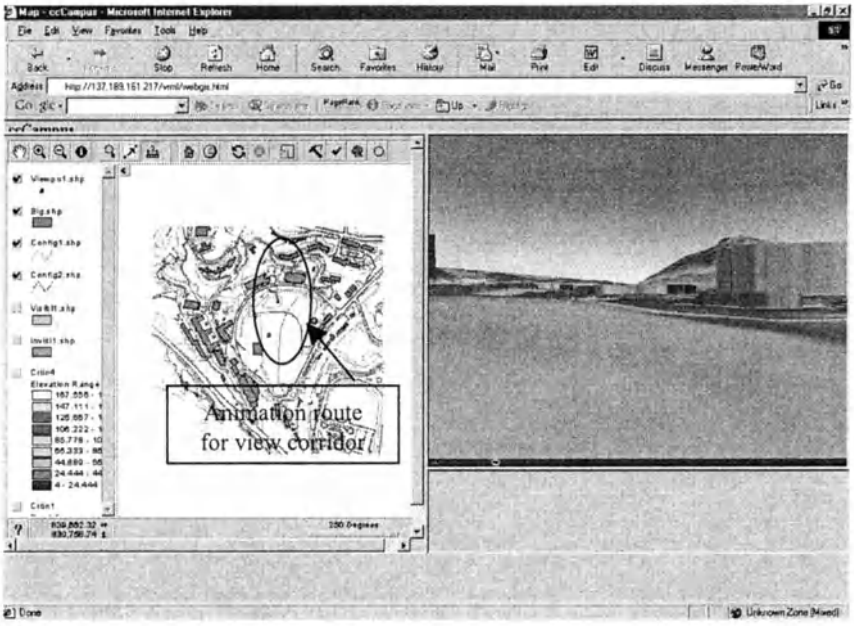


Figure 3. Setting up a tour route in the GIS and triggering the animation in VRML.

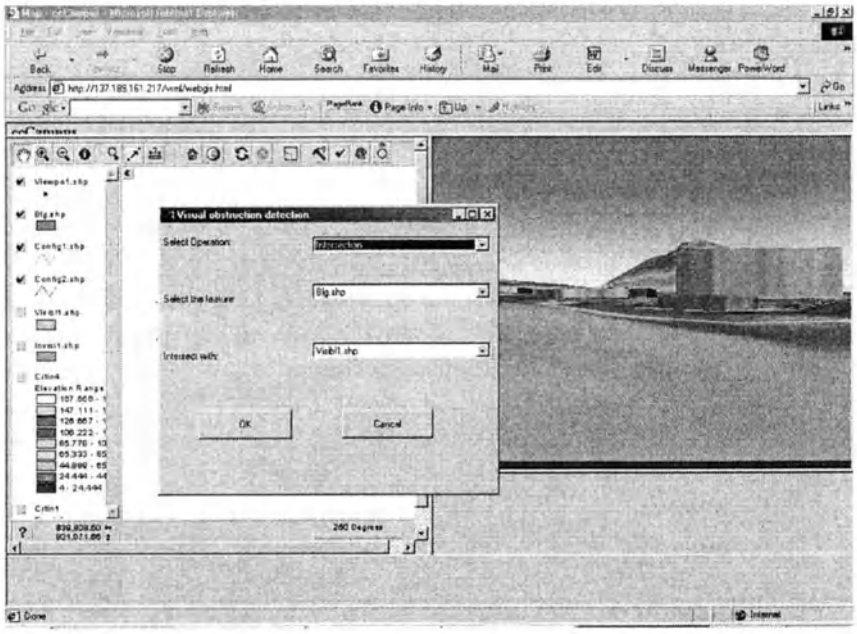


Figure 4. Control panel for intervisibility analysis of interesting features.

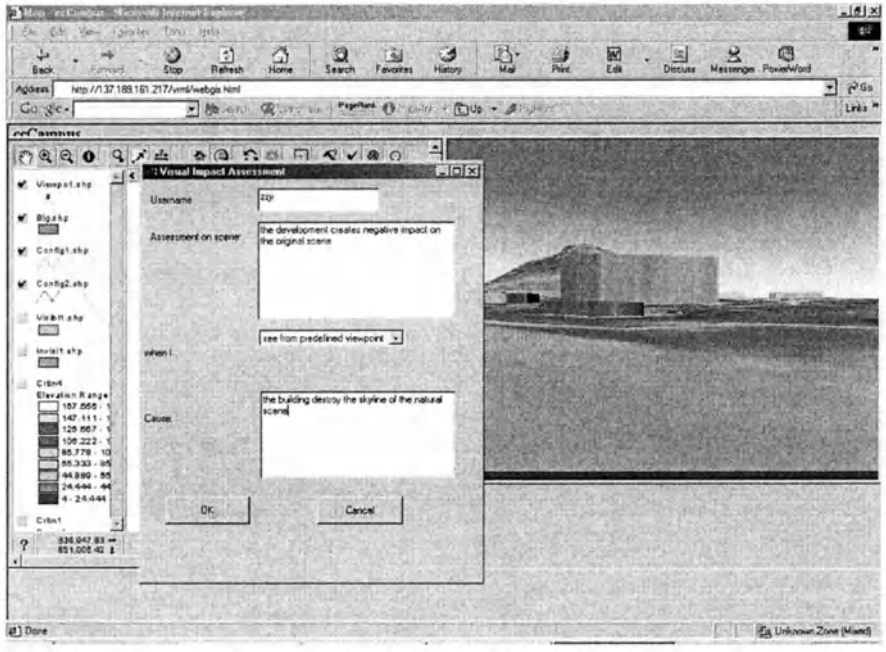


Figure 5. User feedback for visual impact assessment.

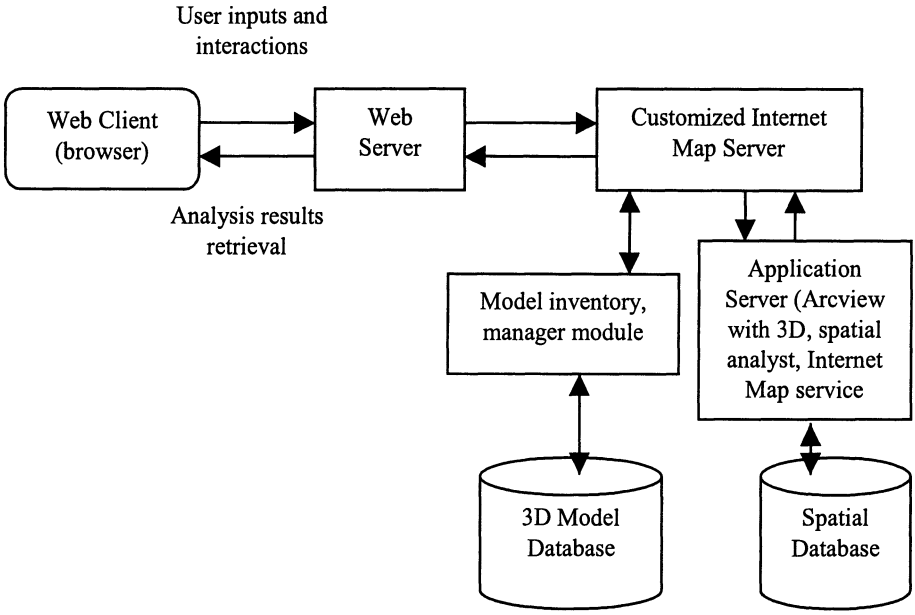


Figure 6. The system's client/server architecture.

3.3 Project Management

The process of urban planning can be divided into several stages. In the first stage of a project, planners interview, poll, survey, distribute questionnaires, or conduct other investigations to infer the customer's requirements and form the project's norm and specifications. In the second stage, limitations are analyzed to weigh the project feasibility. In the third stage, the project is prototyped and simulated to assess its impact on the existing environment. Urban landscape management is particularly concerned with visual impacts.

We introduce a new mechanism to involve public participation in visual impact assessment over the Internet. The process is diagrammed in Figure 7. Participants are empowered to examine a project via GIS and VRML, to assess whether the visual impacts are good or bad, and to state the reasons for their assessments. In the next stage, these assessments are reviewed by experts or other highly respected participants. These may be trained professionals in urban landscape design, or others elected as "senior representatives" by a democratic vote. They evaluate the negative impacts as logged in the system's database and ascertain the root causes. The planners are notified, and they revise their design accordingly to mitigate the negative impacts.

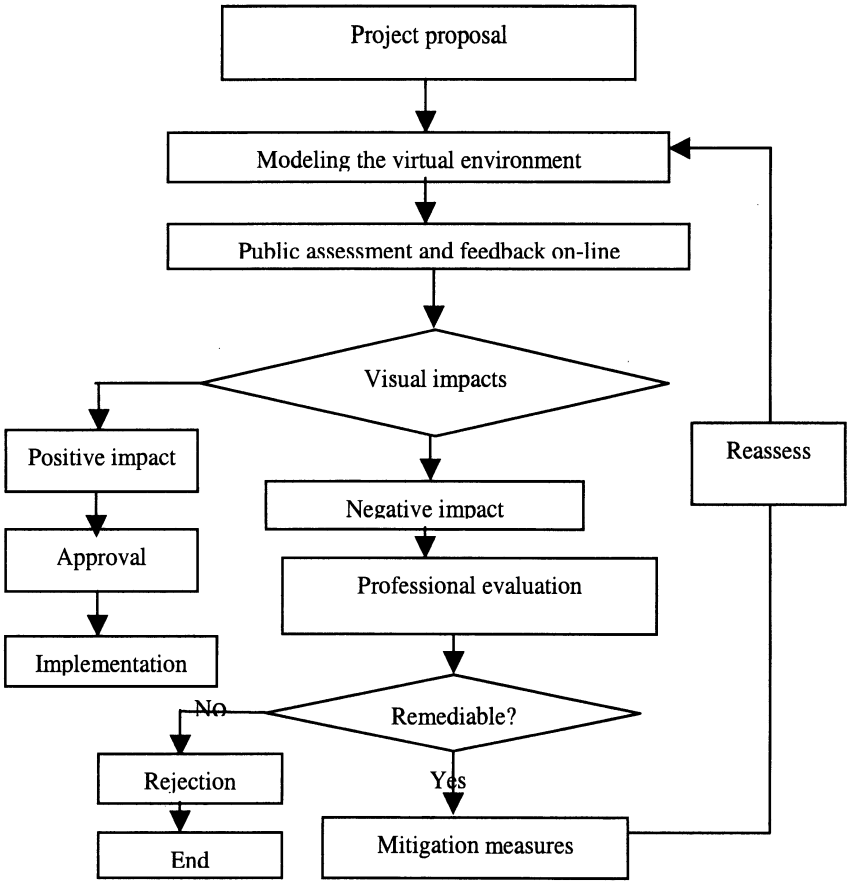


Figure 7. Project management flow diagram.

4. CONCLUSION

Involving public participation in the urban planning process has nearly been an obsolete topic. New technology brings new channels for communication between planners and the public. To facilitate public involvement in visual impact assessment, we integrate a web-based GIS and web-based virtual-reality system. This integrated system allows participants on the Internet to navigate in a virtual environment that represents a proposed development. The system maintains an inventory of VRML objects keyed to the GIS database, and provides tools for quantitative analysis of intervisibility. It logs details of public participation, including view parameters and associated visual impact assessments, for later review by professional designers and planners. To make public participation more

constructive, the log allows reviewers to reconstruct a user's visual experience of the virtual environment and evaluate the root causes of negative assessments.

Web-based public participation has been criticized as an "elite" technology, with an inherent bias for people with Internet access. Furthermore, the user interface design of our experimental system seems much too "professional" in style. How to close the gap between the general public and a sophisticated assessment system on the Internet, to make our system more accessible to common people, is an urgent matter for future research. The management of VRML models is another headache; the simplified inventory and retrieval module shows weakness in the very frequent update of VRML models. This is also an issue for further research and development.

5. REFERENCES

Dearden, P. 1981. Public participation and scenic quality analysis, *Landscape Planning*, 8, p3-19.

Homma, R., K. Iki, M. Morozumi, and T. Morisaki, 1999, "Geographic Information Database for Landscape Evaluation", *Proceedings of the Fourth Conference on Computer Aided Architectural Design Research in Asia*, Shanghai Scientific and Technological Literature Publishing House.

Institute of Environmental Assessment and The Landscape Institute (IEATLI), 1995, *Guidelines for Landscape and Visual Impact Assessment, First Edition*, Chapman & Hall, London.

Al-Kodmany, K., 2000, "Using Web-Based Technologies and Geographic Information Systems in Community Planning", *Journal of Urban Technology*, 7.

Kingston, R., 1998, "Web based GIS for Public Participation Decision Making in the UK", *Empowerment, Marginalisation, and Public Participation GIS*, National Centre for Geographic Information and Analysis, Santa Barbara, California, USA.

Oh, K., 1993, "A Perceptual Evaluation of Computer-based Landscape Simulations", *Landscape and Urban Planning*, 28, p. 201-215.

Lange, E., 1994, "Integration of Computerized Visual Simulation and Visual Assessment in Environmental Planning", *Landscape and Urban Planning*, 30, p. 99-112.

Orland, B., 1992, "Data Visualization Techniques in Environmental Management", *Landscape and Urban Planning*, 21, p. 237-244.

Orland, B., 1994, "Visualization Techniques for Incorporation in Forest Planning Geographic Information Systems", *Landscape and Urban Planning*, 30, p. 83-97.

Hartman, J., and J. Wernecke, 1998, *The VRML 2.0 Handbook: Building Moving Worlds on the Web*, Addision-Wesley Publishing Company.

Wherrett, J. R., 1996a, "Landscape Preference and Perception", *Visualization Techniques for Landscape Evaluation: Literature Review*, http://bamboo.mluri.sari.ac.uk/~jo/litrev/chapters.html

Wherrett, J.R., 1996b, "Visual Impact Assessment", *Visualization Techniques for Landscape Evaluation: Literature Review*, http://bamboo.mluri.sari.ac.uk/~jo/litrev/chapters.html

VR-DIS Research Programme

Design Systems group

B. de Vries, H.H. Achten, M.K.D. Coomans, J. Dijkstra, S. Fridqvist, A.J. Jessurun, J.P. van Leeuwen, M.A. Orzechowski, D.J.M. Saarloos, N.M. Segers and A.A.W. Tan
Eindhoven University of Technology
www.ds.arch.tue.nl

Key words: Interactive design system, Distributed multi disciplinary design, Interactive measurement of user reactions

Abstract: This paper presents a summary of all on-going projects within the VR-DIS research programme at Eindhoven University of Technology.

1. INTRODUCTION

The VR-DIS research programme is the research work done in the Design Systems group of Eindhoven University of Technology. It was established in 1998, at the same time when the former groups of Building Information and Design Methods were fused. The research programme is based on the main premise that interactive and immersive representation of design (knowledge) will best support the design process in the future. The VR-DIS acronyms Virtual Reality – Design Information System and Distributed Interactive Simulation conform to this belief. Not only the design itself, but also interaction with the design, knowledge of the user, and behaviour of the building design should be accessible in a responsive environment.

Organisationally, the VR-DIS research programme is facilitated by the fact that the Department of Architecture and Building of Eindhoven University of Technology has a wide array of disciplines in the building sciences: not only Design Systems, but also Structural Engineering, Building Physics, Urban Design, Architecture, Real Estate Management,

B. de Vries et al. (eds.), Computer Aided Architectural Design Futures 2001, 795–808.
© 2001 *Kluwer Academic Publishers.*

Construction Management, and Building Technology. Therefore, VR-DIS is not only located with the Design Systems group but also in the other research groups as well. In a wider context, VR-DIS is embedded in the DDSS (Design Decision Support Systems) research programme of the Department of Architecture and Building, in which the group closely operates with the Urban Planning group.

In this paper, we will present an overview of running projects. Design support is the first main issue, in which the work by de Vries and Segers is presented. Feature modelling is the main information modelling approach, which receives attention by van Leeuwen, Achten, Coomans, and Fridqvist. The assessment of designs on building and urban level in various ways are then tackled by Dijkstra, Orzechowski, Tan, and Saarloos.

2. DDDOOLZ

Project conducted by B. de Vries

Sketching in 3D seems to be a paradox. The sketch activity is inherently 2D since it is executed in a plane on a flat surface using some drawing device (e.g. a pencil). Three dimensional creation and manipulation of objects presumes the activity being executed in a 3D environment on spatial objects. DDDoolz (www.ds.arch.tue.nl/Research/DDDoolz/) is a new innovative dedicated system for mass study and spatial design in the early design stage. While designing, the model can be inspected and even experienced from any viewpoint. It is a low-end user system in the sense there is almost no learning time and the software will run on standard computers.

Sketching with DDDoolz is like painting blocks in space. Creating new blocks is best described by a 'copy while drag' operation. For determining the copy direction the Face Orientation Method is introduced (Achten and de Vries 2000). Crossing the edge of an existing block while moving the cursor of the blocks' side will result in creating a new block adjacent to the previous selected one. This fairly simple principle gives the user a sketch-like feeling when dragging a block through space. To support 3D perception of the model, the system provides two windows, one to orbit the viewpoint around the model and another, which allows navigating the viewpoint through the model.

DDDoolz is used in the first year's CAAD course next to other CAD tools like AutoCAD. In research DDDoolz is used as a platform to experiment with different input device combinations like a tablet, pressure sensitive pen, trackball, Desk-CAVE, head tracking, etc.

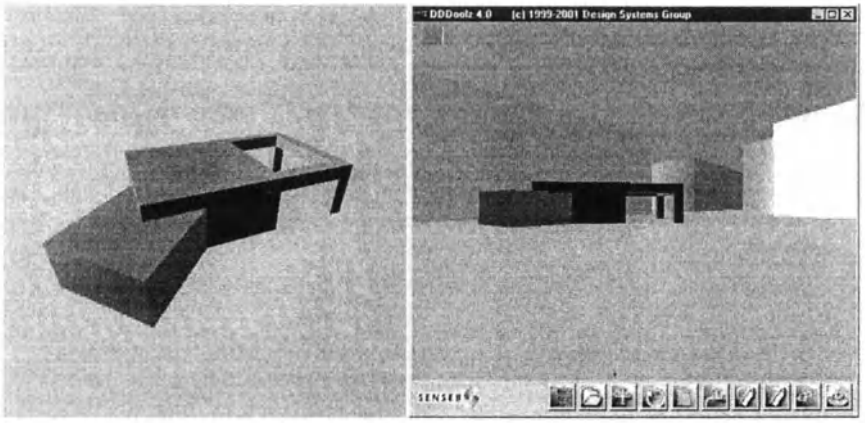


Figure 1. Sketch object and Orbit model (left).
Navigate model and Import environment (right)

3. TOWARDS COMPUTER-AIDED SUPPORT OF ASSOCIATIVE REASONING IN THE EARLY PHASE OF ARCHITECTURAL DESIGN

Project conducted by N.M. Segers

The early phase of the design process is a seemingly chaotic, complex process, involving many methods and representations. Architects try to get a grip on the assignment, and come up with their very first ideas. In general this research (Segers et al. 2000) serves two purposes: understanding architectural design in the early phase and supporting the architect in his/her design process during this phase. We want to improve the architectural design process by means of a design system, which has the advantages that the architect experiences in the traditional design process, but with additional aids. This will be done by a joint effort of three groups, namely the Design Systems Group (DS, Faculty of Building and Architecture), the Computer Graphics Group (CG, Faculty of Mathematics and Computer Science), and the Center for User-System Interactions (IPO).

The data handling of the envisioned system is supported by knowledge on creative cognitive processes (Finke et al. 1992), and knowledge on the design process of the architect. If the data handling of the system resembles how in the human mind ideas and relations are present and used, the design-information can be stored and retrieved in a more or less natural way. The architect manipulates small amounts of data, constructing them together from and to a personal world (Schön 1983); he tends to tell himself stories

(Witt 2000) and imagines how ideas for the future users will work. The basis for constructing a personal world and of telling stories is the personal 'idea space' and the cognitive processes to alter it.

In the system we will make a division between general design-data and personal design-data. The general design knowledge databases or case bases contain information that the architects use from former projects or from other architects. The current design assignment that the architect is working on is represented in the 'idea space'. It consists of a totality of nodes – representing ideas –, links – representing relations between ideas –, and their properties. These properties imply the time and date of creation or use, the design assignment when it was created or used, next to which nodes was the node or link created or used, and content related properties. The envisioned system must be able to tell what kind of ideas and relations it is dealing with, in order to describe the nodes and links.

With this structure, it is possible to show the development of ideas in time, to show the current state of the design and to suggest ways to continue via association. The emphasis of this research will be on the 'idea space' and it's possibilities, i.e. association.

4. FEATURE-BASED MODELLING

Project conducted by J.P. van Leeuwen and A.J. Jessurun

Management of design information is a crucial part of design and decision support systems. The information modelling approach developed in the VR-DIS programme is derived from Feature-based modelling (van Leeuwen 1999). It provides three major capabilities: (1) it allows for extensibility of conceptual schemas, which is used to enable a designer to define new typologies to model with; (2) it supports sharing of conceptual schemas, called type-libraries; and (3) it provides a high level of flexibility that offers the designer the opportunity to easily reuse design information and to model information constructs that are not foreseen in any existing typologies. The latter aspect involves the capability to expand information entities in a model with relationships and properties that are not typologically defined but applicable to a particular design situation only; this helps the designer to represent the actual design concepts more accurately.

The VR-DIS system consists of a growing number of modules for different kinds of functionality in relation with the design task. These modules access the design information through an Application Programming Interface (API) that implements the meta-layer of data definitions in both conceptual models and actual design models. This API has previously been

implemented using an Object-Oriented Database, but is now based on eXtensible Markup Language (XML) (van Leeuwen and Jessurun 2001).

Research in this project currently focuses on Internet-based applications that support designers in the utilisation of distributed libraries of product-information, design-knowledge, case-bases, etc.

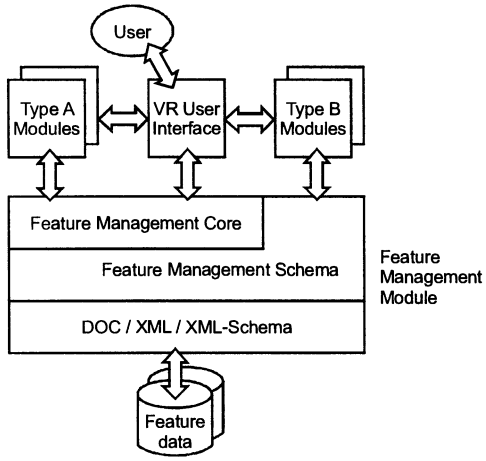


Figure 2. XML Schema based system implementation (van Leeuwen and Jessurun 2001)

5. DESIGN SUPPORT BY FEATURES

Project conducted by H.H. Achten

As stated above, Feature-Based Modelling is the basic information model that is being used and developed in the VR-DIS research programme. One of the main goals is to provide an extensible and flexible approach to information modelling, in order to better support architectural design. The provision of an information modelling paradigm alone is not sufficient to accomplish this, since the formulation itself does not incorporate the dynamics and complexities of everyday design processes. In this project therefore, we are investigating how Feature Models of designs change during the design process. For this purpose, we analyse concrete design cases. The analysis is based on the drawings produced throughout the design process. Each drawing is taken as a step in the process, and described as a Feature Model. We can then describe changes in the Feature Model related to design actions in the design process. Based on this work, which has been reported in (Achten and van Leeuwen 1999; 2000), we can then determine what kind of tools the designer needs when working with Features.

A second track in the research work concerns the implementation of the work on generic representations (Achten 1997). Generic representations are well-defined graphic representation of which the design decisions associated with the drawing can be inferred. Currently, we are investigating how generic representations can be modelled as Features. When this has been done, then other techniques such as Case Based Reasoning can be utilised to support design (Achten 2000).

6. DDDIVER

Project conducted by M.K.D. Coomans

A core part of the VR-DIS research is the innovative, Feature based design-information modelling technique that was developed by van Leeuwen (van Leeuwen 1999). In this modelling technique, the architectural designer is offered a way to control how abstract design data is structured and stored. With this new information modelling technique, we expect that the designers will be better capable of handling the complexity of linking diverse kinds of information involved in a building design process. The drawback of the gained information modelling freedom is that this new technique also puts the responsibility for the content of the CAD database entirely in the hands of the designer. In order to be able to enjoy the design freedom fully and at the same time handle the responsibility over the design database, a computer tool is needed that shows the precise content of the database, and that is easy and quick to interact with. A 3D graphical tool was developed to meet these demands, called DDDiver.

DDDiver visualises the Feature database in two interactive graphs (see figure). One graph shows the Feature *model* (an actual design), and another the Feature *Type library*. Features and Feature Types can be inspected and manipulated on transparent layers. An interactive data exploration mechanism was developed that dynamically moves data objects to layers laying more at the back, in order to maintain context-data in view during data navigation (Coomans 2001).

DDDiver can be controlled by a standard mouse and keyboard. Optionally, these standard devices can by complemented with head tracking and tailor-made manipulation devices, in order to obtain a so-called *fish-tank VR* set-up. These additional input devices facilitate improved depth perception and easier object manipulation.

User experiments are conducted to validate the tool's performance on a larger audience and to guide the further developments.

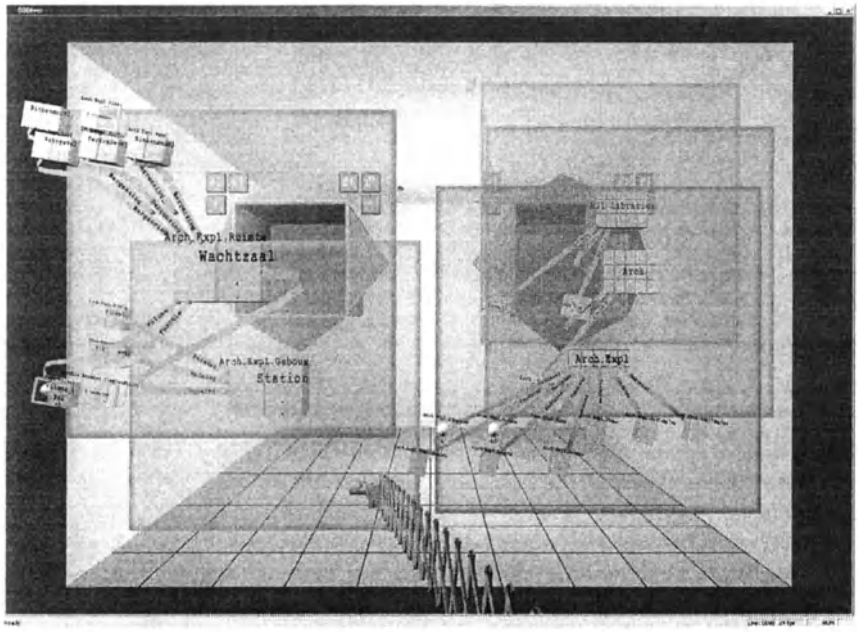


Figure 3. The interactive visualisation of the Feature model (on the left) and the Feature Type library (on the right) in DDDiver

7. IMPLEMENTATION AND APPLICATION OF FEATURE TYPE RECOGNITION

Project conducted by S. Fridqvist

The Feature Based Modelling (FBM) system addresses the most foundational questions of how to construct information systems for the construction and facility management industry (van Leeuwen 1999). Several benefits that are possible to gain with the FBM system, however, require an additional technique, Feature Type Recognition (FTR) (van Leeuwen and de Vries 2000).

While the FBM paradigm gives great flexibility and expressiveness, it also opens the risk to create models with little or no reference to common practices. This will of course render such models unusable for communication of design information. The FTR technique can decrease this risk by automatically or semi-automatically link the user-defined classes to approved class libraries, thereby giving models with user-defined classes the necessary basis for communication.

Additionally, a FBM system augmented with FTR could assist designers and other professionals in several common tasks. Since FBM is a general modelling technique, several kinds of information might be modelled using FBM. Feature type recognition could then be used to analyse different aspects of design models, to search in common depositories for missing model information, and to translate models between different national or industrial standards of modelling.

This sub-project within the VR-DIS frame will add Feature type recognition to the FBM system, gaining from the main researcher's earlier experiences at another institution (Fridqvist 2000). It will also investigate and demonstrate FTR by applying it to one or several design tasks.

8. AMANDA – A MULTI AGENT SYSTEM FOR NETWORK DECISION ANALYSIS

Project conducted by J. Dijkstra

Architects and urban planners are often faced with the problem to assess how their design or planning decisions will affect the behaviour of individuals. Various performance indicators are related to the behaviour of individuals in particular environments. One way of addressing this problem is to develop models, which relate user behaviour to design parameters. For example, models of pedestrian behaviour have been developed to support design decisions related to the location of facilities, parking policies, etc.

Graphical representations and 3D simulations may be powerful tools to assess design parameters in terms of such behaviour. Therefore, we formulated a research project that aims at exploring the possibilities of developing such a tool in a virtual reality environment using multi-agent simulation and cellular automata technology. The particular problem of reference is to assess planning and design decisions related to the design of shopping malls in terms of pedestrian behaviour, which in turn will influence the performance of the mall.

The purpose of this research is to describe a multi-agent system that can be used for network decision analysis. The term network decision analysis is used to encompass all design and decision problems that involve predicting how individuals make choices when moving along a network (corridors in a building, streets, highways, etc.). This model is based on cellular automata and multi-agent technology. The underlying idea is to model how agents move in a particular 3D (or 2D) environment. A cellular automata model of an environment in which space is represented as a uniform lattice of cells with local states, subject to a uniform set of rules constitutes the base of the system. These rules compute the state of a particular cell as a function of

previous state and the states of the adjacent cells. An extension of the basic model of cellular automata allows cells to be influenced by more than their immediate neighbours; state changes may depend on the aggregate effect of the states of all other cells, or some proportion of them. Another extension is to build models in which cells preserve state information and calculate their next state on the basis of their neighbours and their own history of state changes. Agent technology will be implemented to build a framework for multi-agent simulation. Agents represent objects or people moving over the network. Each agent is located in a simulated space, based on the cellular automata grid. Agents positioned within an environment need sensors to perceive their local neighbourhood and some means with which to affect the environment. In Dijkstra *et al* (2000) this research has been described.

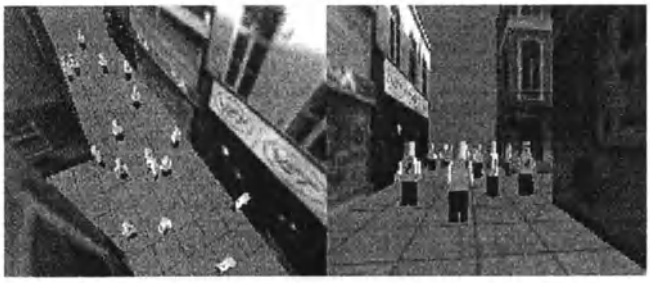


Figure 4. Movement Visualisation

9. MEASUREMENTS OF USER'S SATISFACTION IN VR FOR BUILDING ENVIRONMENT

Project conducted by M.A. Orzechowski

The key notion of this research is to measure user's satisfaction unobtrusively by having them interactively create their preferred design. We will develop and test a VR System for interactive measurement of user satisfaction. Furthermore, we intend to improve on the method of conjoint analysis for predicting user satisfaction on design alternatives.

Artificial intelligence technology will be introduced by employing the agent technology, aiming for a truly user-friendly VR System for non-architectural designers. Modification and manipulation of elements of the design will be possible in a Virtual World (VW) by browsing through a library of design elements (alternatives). Changes are immediately visually represented. The user is able to judge the outcome and respond on-the-spot. This interaction is the main advantage over text-based and image-based experiments to

measure user satisfaction. Agent tools will be used to collect process data and analyze information provided by a user.

An exploration of possibilities to use VR to interactively change architectural design is accomplished via MuseV, a system developed for this purpose.

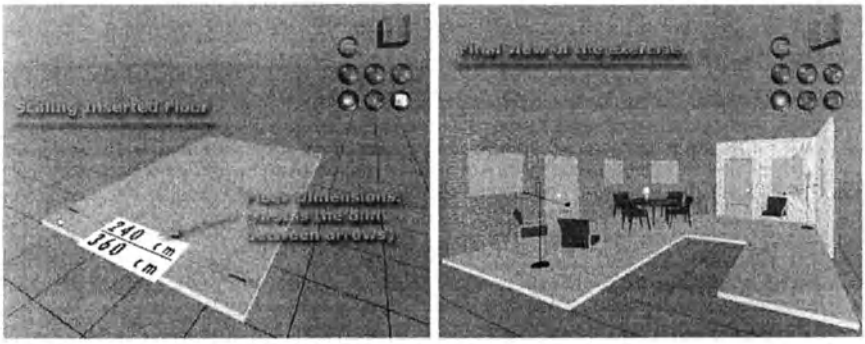


Figure 5. Muse V screen shots

10. MASQUE, A MULTI-AGENT SYSTEM FOR SUPPORTING THE QUEST FOR URBAN EXCELLENCE

Project conducted by D.J.M. Saarloos

A local land use plan gives regulations, in both quantitative and qualitative means, for the types of land uses to be applied in a strictly bounded area. During the making of such a plan an urban planner is confronted with many problems he cannot properly solve on his own. For this reason, he is forced to consult different sources of knowledge, being mainly persons and software tools. A hampering factor is that different sources often need to be consulted simultaneously in an interrelated manner. Urban planning practice has shown that, especially when multiple persons are involved in such multi-source consultations, the planning process can be slowed down dramatically due to the time it will take to simply arrange a meeting and to reach overall agreement. Additionally, when multiple software tools are to be consulted, the urban planner will be discouraged by the fact that these tools are often separately implemented, making it hard to combine them. Due to the described problems, the urban planner often turns to intuition-based decision-making, a rather questionable way of working.

What could really improve the practice of making local land use plans is a computer system incorporating all needed sources of knowledge. MASQUE (Saarloos et al. 2000) will become such a system, built up out of the following four components: Geographical Information System (GIS); Virtual Reality (VR); Multi-Agents (MA); and an extensible set of design and decision support (DDSS) tools. The sources of knowledge will be implemented by using multiple software agents, making a distinction between basically three types: *Interface Agents* taking care of assisting the user and contacting other agents; *Tool Agents* representing modelling experts by offering support on the use of specific decision support tools; and *Domain Agents* representing the multidisciplinary experts taking part in the planning process. By enabling the agents to communicate, the system will offer full support for multi-source consultations.

11. THE RELIABILITY AND VALIDITY OF INTERACTIVE VIRTUAL REALITY COMPUTER EXPERIMENTS

Project conducted by A.A.W. Tan

This research (Tan et al. 2000) explores on how to employ virtual reality in setting up interactive computer experiments as a means to probe the attitudes, motivations, reasoning and principles underlying travel decision-making. The concept of the problem of scheduling and implementing daily activities is viewed as a complex decision making process where individuals develop strategies to realize their needs and goals. A transportation system serves to provide the necessary links between locations of activities.

By establishing the combination of an interview with the technology of virtual reality display equipment in an interactive survey procedure we postulate that this will enable us to collect data that replicates realistic feedback to responses in not-yet-existing scenarios of a transportation system and/or in the introduction of new conditions of the travel options as opposed to real situations.

The environment representation used in virtual reality can incorporate features of the activity framework including the deductive logic of the time-space format that aids in the comprehension of circumstances, acts as a (passive visual) prompt, and serves as an aide-memoire. An essential idea is to automatically collect contextual information about daily human activities and their characteristics, and to use this information to help the later recall of past activities. Some aspect of simulating the actual physical action of walking, cycling, or driving creates the experience of movement. Behaviour in the virtual environment can be monitored by an observing agent that

collects information about the interactions amongst the various objects in the virtual world. Agent tools generating statistics about subjects' movement and choices, their response performance will be used, as well as recording the number of inconsistencies and unaccounted time. We hope to infer from the experiential information "extracted" from the subjects their reasoning, attitudes, and motivation of decision-making in the context of activity scheduling.

12. CONCLUSION

From 1998, the VR-DIS research programme has been running in the Design Systems group. In the meantime, already some prototypes and methods have been developed (see Table 1).

Table 1. Products and methods in VR-DIS

Products	Methods
Feature Manager	Feature Based Modelling. Flexible and extensible data information model for design.
CRB system	Generic representations. Analysis method for capturing design decisions. Retrieval of cases based on design drawings.
Exspect simulation	Message exchange model. Simulation of the information exchange process between participants in a building project.
Feature view / DDDiver	Feature visualisation and manipulation in Virtual Reality.
DDDoolz	Face Orientation Method. The face of blocks as plane of reference for generating new blocks.
VIP (with IPO) / E3DAD	Augmented reality system for collaborative design. Design support for associative reasoning in design.
3D Realtime Constraint solver (with W&I)	Geometric constraints. Constraint definition between 3D objects. Propagation and maintenance in a designerly fashion.
Blocks (with W&I)	Implicit geometric relations. Relations and actions defined and used without explicitly stating them.
ACAD-3DS-WUP Walker	VR cycle. Fast cyclic method to incorporate VR in the design process.
WEDA & ILSA (with Building Physics)	CBR and KBS. Reasoning and design support in the areas of lighting design and workplace comfort.

13. ACKNOWLEDGEMENTS

Many research groups inside and outside the Eindhoven University of Technology cooperate in the VR-DIS research programme, especially the Center for User-System Interactions (IPO), The Computer Graphics group of the Faculty of Mathematics and Computer Science, The Building Physics group and The Urban Planning group of the Faculty and Architecture, Building and Planning.

14. REFERENCES

Achten, H.H., 1997, *Generic Representations – An Approach to Modelling Procedural and Declarative Knowledge in Architectural Building Types*, Ph.D. Thesis, Eindhoven University of Technology, Eindhoven.

Achten, H.H., 2000, "Design Case Retrieval by Generic Representations", in: Gero, J. (ed.), *Artificial Intelligence in Design 2000, Worcester, USA*, Kluwer Scientific Publishers, Dordrecht.

Achten, H.H. and J.P. van Leeuwen, 2000, "Towards Generic Representations of Designs Formalised as Features", in: Timmermans, H.J.P. and B. de Vries (eds.), *Design and Decision Support Systems in Architecture – Proceedings of the 5th International Conference, The Netherlands*, Eindhoven University of Technology, p. 1-14.

Achten, H.H. and J.P. van Leeuwen, 1999, "Feature-Based High Level Design Tools – A Classification", in: Augenbroe, G. (ed.), *Computers in Building – Proceedings of the 8th International Conference on Computer Aided Architectural Design Futures*, Georgia, Atlanta, Kluwer Scientific Publishers, Dordrecht, p. 275-290.

Achten, H.H. and B. de Vries, 2000, "DDDoolz: A Virtual Reality Sketch Tool for Early Design", in: *Proceedings of The Fifth Conference on Computer Aided Architectural Design Research in Asia*, Singapore, p.451-460.

Coomans, M.K.D., 2001, "DDDiver: 3D Interactive Visualization of Entity Relationships", in *Proceedings of the Joint Eurographics and IEEE TCVG Symposium on Visualization,* Ascona, Switzerland, Springer Verlag, Wien (forthcoming).

Dijkstra, J., H.J.P. Timmermans, and B. de Vries, 2000, "Towards a multi-agent system for visualising simulated behaviour within the built environment", in: *Design and Decision Support Systems in Architecture – Proceedings of the 5th International Conference, The Netherlands,* Eindhoven University of Technology, p. 101-117.

Finke, R.A., T.B. Ward, and S.M. Smith, 1992, *Creative Cognition*, MIT Press. Cambridge MA.

Fridqvist, S., 2000, *Property-Oriented Information Systems for Design*, PhD thesis, Department of Construction and Architecture, Lund Institute of Technology.

van Leeuwen, J.P., 1999, *Modelling Architectural Design Information by Features*, PhD Thesis, Eindhoven University of Technology, Eindhoven.

van Leeuwen, J.P., and A.J. Jessurun, 2001, "XML for Flexibility and Extensibility of Design Information Models", in: *Proceedings of CAADRIA 2001*, Sydney (forthcoming).

van Leeuwen, J.P., and B. de Vries, 2000, "Modelling with Features and the formalisation of early design knowledge", in: *Proceedings of the Third European Conference on Product and Process Modeling in the building and related industries*, Lisbon, Portugal, p. 167-176

Orzechowski, M.A., H.J.P. Timmermans and B. de Vries, 2000, "Measuring user satisfaction for design variations through virtual reality", in: Timmermans, H.J.P. and B. de Vries (eds.), *Design and Decision Support Systems in Architecture – Proceedings of the 5th International Conference, The Netherlands*, Eindhoven University of Technology, p. 278-288.

Saarloos, D.J.M., T.A. Arentze, A.W.J. Borgers, and H.J.P. Timmermans, 2001, "Introducing Multi-Agents in an integrated GIS/VR system for supporting urban planning and design decisions", in: *Proceedings of the 7th international Conference on Computers in Urban Planning and Urban Management, Hawaii, USA* (forthcoming).

Segers, N.M., B. de Vries, H.H. Achten, and H.J.P. Timmermans, 2000, "A comparison of computer-aided tools for architectural design", in: Timmermans, H.J.P. and B. de Vries (eds.), *Design and Decision Support Systems in Architecture – Proceedings of the 5th International Conference, The Netherlands*, Eindhoven University of Technology, p. 325-340.

Schön, D, 1983, *The Reflective Practitioner*, New York, Basic Books.

Tan, A.A.W., H.J.P, Timmermans, and B. de Vries, 2000, "Investigation of Human Behavior in Virtual Environments", in: *Proceedings of the VWsim'00 Conference, 2000 Virtual Worlds and Simulation Conference, San Diego, California* (forthcoming).

Witt, T., 2000, "Indecision in quest of designing", in: Timmermans, H.J.P. and B. de Vries (eds.), *Design and Decision Support Systems in Architecture – Proceedings of the 5th International Conference, The Netherlands*, Eindhoven University of Technology, p. 423-431.

AUTHOR INDEX

KEYWORD INDEX