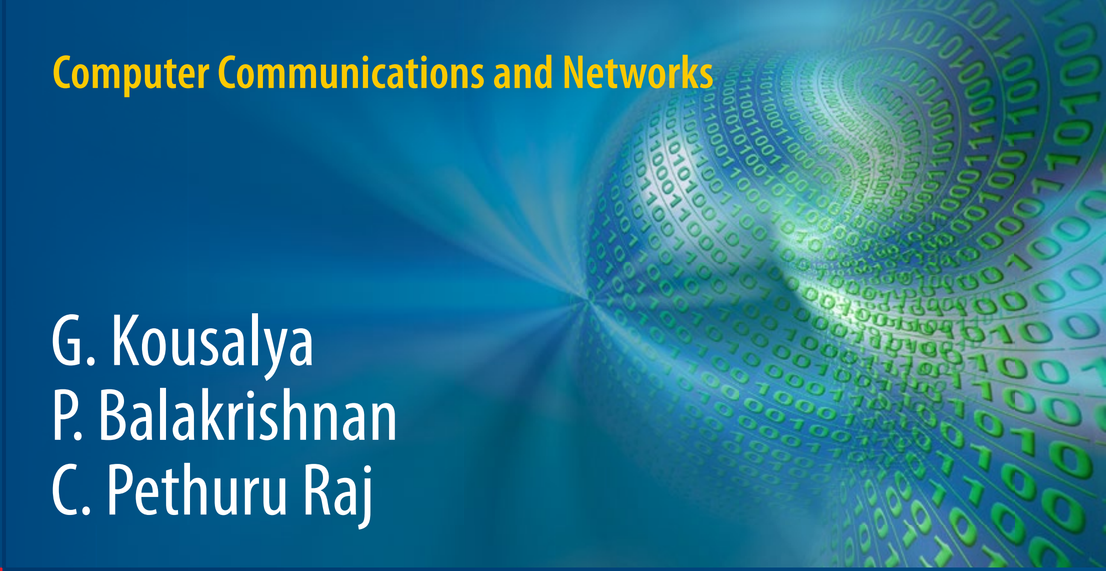
Computer Communications and Networks



G. Kousalya
P. Balakrishnan
C. Pethuru Raj

Automated Workflow Scheduling in Self-Adaptive Clouds

Concepts, Algorithms and Methods

Springer

Computer Communications and Networks

Series editor
A.J. Sammes
Centre for Forensic Computing
Cranfield University, Shrivenham Campus
Swindon, UK

The **Computer Communications and Networks** series is a range of textbooks, monographs and handbooks. It sets out to provide students, researchers, and non-specialists alike with a sure grounding in current knowledge, together with comprehensible access to the latest developments in computer communications and networking.

Emphasis is placed on clear and explanatory styles that support a tutorial approach, so that even the most complex of topics is presented in a lucid and intelligible manner.

More information about this series at http://www.springer.com/series/4198

G. Kousalya • P. Balakrishnan • C. Pethuru Raj

Automated Workflow Scheduling in Self-Adaptive Clouds

Concepts, Algorithms and Methods

Springer

G. Kousalya
Coimbatore Institute of Technology
Coimbatore, India

P. Balakrishnan
SCOPE, VIT University
Vellore, India

C. Pethuru Raj
Reliance Jio Cloud Services (JCS)
Bangalore, India

ISSN 1617-7975 ISSN 2197-8433 (electronic)
Computer Communications and Networks
ISBN 978-3-319-56981-9 ISBN 978-3-319-56982-6 (eBook)
DOI 10.1007/978-3-319-56982-6

Library of Congress Control Number: 2017941762

Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer International Publishing AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Foreword

Enterprise-class software applications are steadily embracing the cloud idea in order to succulently reap all the originally envisaged cloud benefits. Due to the on-demand utility and elastic nature of virtualized and containerized infrastructures that are the real hallmark of any cloud environments (private, public, and hybrid), scores of mission-critical workloads are being accordingly modernized (cloud-enabled) and migrated to cloud environments to be delivered with all alacrity and authentication to worldwide clients and consumers. On the other hand, of late, there are cloud-native applications gaining prominence. There are business, technical, embedded, social, operational, transactional, and analytical applications efficiently running on cloud hosts. The cloud paradigm is definitely on the fast track. That is, we are all set to experience software-defined cloud environments in the days ahead. Precisely speaking, clouds emerge as the one-stop IT solution for hosting, delivering, billing, monitoring, measuring, managing, and maintaining all kinds of simple as well as complex workloads.

As I see, this book is all about expressing and exposing the various automated workflow scheduling algorithms and approaches for various process-centric cloud-based applications. Workflow is typically described by a Directed Acyclic Graph (DAG) in which each computational task is represented by nodes and each data/control dependency between tasks is annotated through edges that intrinsically connect nodes. Workflow scheduling is therefore recognized as one of the most vital requirements for hosting workflow-centric applications in cloud environments. There are quality of service (QoS) constraints such as the timeliness, throughput, minimal cost, minimal makespan and maximal resource utilization, etc. The other widely articulated and accentuated challenge is the efficient resource utilization and optimizing the total execution time (makespan) of the workflow.

Having understood the intricacies of workflow applications and the state-of-the-art workflow/task/job scheduling algorithms, the authors of this comprehensive yet compact book have clearly detailed the enterprise-grade software applications and their scheduling needs in cloud environments. This book covers most of the topics that are needed for cloud consultants, architects, and administrators. The cloud service providers (CSPs) across the globe are leveraging a variety of scheduling algorithms in order to enhance resource utilization of highly virtualized IT environments for bringing down the cloud operational costs. I am sure that this book is a must for professionals who are manning next-generation software-defined cloud environments. Finally, research students, scholars, and scientists are bound to be benefited immensely through this book.

Dr. K.R. Murali Mohan

Scientist-G & Head, Interdisciplinary
Cyber Physical Systems Division (ICPS),
Department of Science and Technology,
Technology Bhavan, New Mehrauli Road,
New Delhi, India

Preface

Two things are clearly noteworthy here. There is a heightened heterogeneity of cloud IT resources, whose distributed nature also is on the climb. Further on, the multiplicity of software applications leveraging various cloud resources is steadily growing. All these points to the fact that the IT development, deployment, delivery, and management complexities are bound to escalate sharply in the days ahead. There are pioneering technologies, enabling tools, and advanced algorithms emerging and evolving fast, and if they are applied correctly, the threatening complexity of new-generation IT environments is bound to decline substantially. Academic professors and IT industry professionals across the globe hence work collaboratively to unearth a bevy of workable solutions for the abovementioned IT challenges. Optimized and organized scheduling of jobs/tasks/workflows of process-aware, service-oriented, event-driven, cloud-centric, and enterprise-scale applications is one forward-looking complexity-mitigation requirement for the impending cloud IT era. This book is expertly crafted to describe the various workflow scheduling algorithms and approaches, which are becoming indispensable to bring a kind of sanity and also to run all kinds of software applications (cloud-enabled and cloud-native) by efficiently leveraging different and distributed cloud resources.

Chapter 1 (Stepping into the Digital Intelligence Era) is to talk about the digitization technologies and how myriads of digitized entities and elements are being systematically realized and deployed in our daily environments. Further on, the chapter digs deeper and describes how the connected era emerges and evolves, how the massive amount of data getting generated are subjected to a variety of investigations to squeeze out viable and venerable insights, and finally how the knowledge extracted is delivered to facilitate correct decision-making and action in time. The chapter ends with how digitization and the paradigm of cognitive computing converge with one another in order to make the path smooth for the forthcoming era of digital intelligence.

Chapter 2 (Demystifying the Traits of Software-Defined Cloud Environments (SDCEs)) is specially crafted to tell all about the cloud journey. We have started with the brief of age-old server virtualization and then proceeded to explain how other cloud resources especially storage and network are getting virtualized. The brewing trend is to have completely virtualized IT environments by appropriately leveraging cloud technologies. There are virtual machine monitors (VMMs), integration tools, orchestration engines for automated provisioning and software deployment, service brokers for multi-cloud solutions, integrated monitoring, measurement and management systems, job schedulers, capacity planning and configuration tools, a plenty of security algorithms and approaches, and other automated solutions to have next-generation cloud environments. The readers can find the details in the second chapter.

Chapter 3 (Workflow Management Systems) is incorporated to give a detailed explanation of workflow management systems. Today's scientific applications require a tremendous amount of computation-driven as well as data-driven supported resources. Typically scientific applications are represented as workflows. The workflow management systems are designed and developed to depict the workflows of complex nature. The workflow management systems are able to reliably and efficiently coordinate among various resources in a distributed environment. This chapter describes various workflow management software solutions such as Kepler, Taverna, Triana, Pegasus, and Askalon. The architecture and functionalities of these workflow management systems are explained in a lucid manner.

Chapter 4 (Workflow Scheduling Algorithms and Approaches) is to explain the nitty-gritty of various workflow scheduling algorithms. Cloud infrastructures typically offer access to boundless virtual resources dynamically provisioned on demand for hosting, running, and managing a variety of mission-critical applications. Efficient scheduling algorithms become mandatory for automated operations of distributed and disparate cloud resources and workloads. The resource scheduling is a dynamic problem; it is associated with on-demand resource provisioning, fault tolerance support, hybrid resource scheduling with appropriate quality of service, and considering time, cost, and budget. This chapter provides the details about various automated solutions for workflow scheduling and also a comprehensive survey of various existing workflow scheduling algorithms in the cloud computing environment.

Chapter 5 (Workflow Modeling and Simulation Techniques) is to detail the prominent and dominant workflow modeling and simulation techniques and tips. Modeling and simulation of scientific workflow play a vital role in resource allocation in a distributed environment. Simulation is one of the methods to solve the complex scientific workflows in distributed environment. There are many scientific workflow simulation software frameworks that are available for grid and cloud environment. WorkflowSim is an open-source simulator. WorkflowSim Simulator extends the existing CloudSim Simulator. The architecture, components, and scheduling algorithms used and also the simulation results are explained for the CloudSim Simulator and WorkflowSim Simulator.

Chapter 6 (Execution of Workflow Scheduling in Cloud Middleware) is to converge the workflow capabilities in cloud environments. Many scientific applications are often modeled as workflows. The data and computational resource requirements are high for such workflow applications. Cloud provides a better solution to this problem by offering a promising environment for the execution of these workflows.

As it involves tremendous data computations and resources, there is a need to automate the entire process. The workflow management system serves this purpose by orchestrating workflow task and executes it on distributed resources. Pegasus is a well-known workflow management system that has been widely used in large-scale e-applications. This chapter provides an overview of the Pegasus Workflow Management System and describes the environmental setup with OpenStack, creation, and execution of workflows in Pegasus and discusses the workflow scheduling in the cloud with its issues.

Chapter 7 (Workflow Predictions Through Operational Analytics and Machine Learning) is an important one for this book. Data analytics is the widely recognized mechanism to squeeze out important information out of historical as well as current data heaps. The advancements in the fields of operational analytics and machine learning (ML) clearly could foretell everything to accurately predict workflows. Increasingly workflow execution employs predictive analytics to extract significant, unidentified, as well as precious insights from several stages of execution. Further, the operational analytics integrates these valuable insights directly into the decision engine which enables analytical as well as machine learning-driven decision-making for an efficient workflow execution. This chapter highlights several analytical and machine learning approaches that are practiced in workflow predictions. Additionally, it explains the significance of a hybrid approach which includes both analytical and machine learning models for workflow prediction. Finally, it describes the hybrid approach employed in PANORAMA architecture using two workflow applications.

Chapter 8 (Workflow Integration and Orchestration, Opportunities and Challenges) is prepared and presented in order to explain how workflow orchestration is being performed. Workflow orchestration is a method which smartly organizes the enterprise function with the application, data, and infrastructure. The applications, as well as their infrastructure, can be dynamically scaled up or down using orchestration. On the contrary, the integration enables the development of new applications with the capability to connect to any other application through specified interfaces. In this chapter, firstly, the opportunities and challenges in workflow orchestration and integration are explained. Following that, BioCloud, an architecture that demonstrates the task-based workflow orchestration using two bioinformatics workflows, is explained in detail.

Chapter 9 (Workload Consolidation Through Automated Workload Scheduling) illustrates how workload consolidation and optimization lead to heightened resource utilization. Workload consolidation is an approach to enhance the server utilization by grouping the VMs that are executing workflow tasks over multiple servers based on their server utilization. The primary objective is to optimally allocate the number of servers for executing the workflows which in turn minimize the cost and energy of data centers. This chapter consolidates the cost- and energy-aware workload consolidation approaches along with the tools and methodologies used in modern cloud data centers.

Chapter 10 (Automated Optimization Methods for Workflow Execution) deals with how various optimization methods guarantee optimal execution of workflows.

Workflow optimization is an approach to enhance the speed, robustness, and compactness of workflows by exploiting their structure, runtime, and output. This chapter initially highlights the significance of workflow optimization along with different possible levels of optimization. Further, it outlines the Taverna optimization framework over single and distributed infrastructure together with the optimization plugins that is validated using two scientific workflow executions.

Chapter 11 (The Hybrid IT: The Characteristics and Capabilities) is to give an idea of the emerging hybrid IT domain. With the faster adoption of the cloud idea across industry verticals with all the elegance and the enthusiasm, the traditional IT is bound to enlarge its prospects and potentials. This is a kind of new IT getting enormous attention and garnering a lot of attraction among business executives and IT professionals lately. The systematic amalgamation of the cloud concepts with the time-tested and trusted enterprise IT environment is to deliver a bevy of significant advantages for business houses in the days ahead. This model of next-generation computing through the cognitive and collective leverage of enterprise and cloud IT environments is being touted as the hybrid IT. This chapter is specially crafted for digging deep and describing the various implications of the hybrid IT.

Coimbatore, India G. Kousalya
Vellore, India P. Balakrishnan
Bangalore, India C. Pethuru Raj

Acknowledgments

G. Kousalya
I express my sincere and heartfelt gratitude to my beloved father Thiru R. Govardhanan for his selfless support and motivation in every walk of my life. In remembrance of his enthusiasm and determination, I wholeheartedly dedicate this book to my father.

P. Balakrishnan
First of all, I convey my heartfelt thanks to Wayne Wheeler and Simon Rees of Springer for providing us this opportunity. Besides, I thank my co-authors for their helping hand and valuable suggestions in shaping up the book. Above all, I must thank my beloved parents (Mr. M. Ponnuraman and Ms. P. Parvathy) who taught me the power of hard work. Finally, I thank my brother (Dr. P. Selvan) and sister (Dr. P. Selvi) for their continuous and consistent moral support.

C. Pethuru Raj
I express my sincere gratitude to Wayne Wheeler and Simon Rees of Springer for immensely helping us from the conceptualization to the completion of this book. I need to remember my supervisors Prof. Ponnammal Natarajan, Anna University, Chennai; Prof. Priti Shankar (late), Computer Science and Automation (CSA) Department, Indian Institute of Science (IISc), Bangalore; Prof. Naohiro Ishii, Department of Intelligence and Computer Science, Nagoya Institute of Technology; and Prof. Kazuo Iwama, School of Informatics, Kyoto University, Japan, for shaping my research life. I express my heartfelt gratitude to Mr. Thomas Erl, the world's top-selling SOA author, for giving me a number of memorable opportunities to write book chapters for his exemplary books. I thank Sreekrishnan, a distinguished engineer in IBM Global Cloud Center of Excellence (CoE), for extending his moral support in completing this book.

I, at this point in time, recollect and reflect on the selfless sacrifices made by my parents in shaping me up to this level. I would expressly like to thank my wife (Sweetlin Reena) and sons (Darren Samuel and Darresh Bernie) for their perseverance as I have taken the tremendous and tedious challenge of putting the book together. I thank all the readers for their overwhelming support for my previous books. Above all, I give all the glory and honor to my Lord and Savior Jesus Christ for His abundant grace and guidance.

Contents

1 Stepping into the Digital Intelligence Era 1
1.1 Introduction 2
1.2 Elucidating Digitization Technologies 2
1.2.1 Why Digitization? 4
1.3 The Internet of Things (IoT)/Internet of Everything (IoE) 6
1.4 Real-Time, Predictive, and Prescriptive Analytics 10
1.5 Envisioning the Digital Universe 12
1.6 Describing the Digitization-Driven Big Data World 13
1.7 The Cloud Infrastructures for the Digitization Era 16
1.8 Integrated Platform for Big Data Analytics (Dr. Barry Devlin [1]) 19
1.9 Conclusion 22
References 22

2 Demystifying the Traits of Software-Defined Cloud Environments (SDCEs) 23
2.1 Introduction 23
2.2 Reflecting the Cloud Journey 25
2.2.1 Elucidating the Cloudification Process 26
2.2.2 The IT Commoditization and Compartmentalization 28
2.3 Visualizing the Future 30
2.4 The Emergence of Software-Defined Cloud Environments (SECEs) 32
2.5 The Major Building Blocks of Software-Defined Cloud Environments (SDCEs) 34
2.5.1 Network Virtualization 35
2.5.2 Network Functions Virtualization (NFV) 36
2.5.3 Software-Defined Networking (SDN) 37
2.5.4 The Key Motivations for SDN 40
2.5.5 The Need of SDN for the Cloud 41

2.6 The Distinct Benefits of Software-Defined Networking 42
2.7 Accentuating Software-Defined Storage (SDS) 44
2.8 The Key Characteristics of Software-Defined Storage (SDS) 47
2.8.1 Software-Defined Wide Area Networking (SD-WAN) 48
2.9 The Key Benefits of Software-Defined Cloud Environments (SDCEs) 49
2.10 Conclusion 53
References 53

3 Workflow Management Systems 55
3.1 Introduction 55
3.2 Workflow Management System 56
3.3 Kepler 56
3.4 Taverna 58
3.5 Triana 60
3.6 Pegasus 61
3.7 ASKALON 62
3.8 Conclusion 64
References 64

4 Workflow Scheduling Algorithms and Approaches 65
4.1 Introduction 65
4.2 Workflow Model 67
4.3 Static Workflow Scheduling 69
4.4 Dynamic Workflow Scheduling 70
4.5 Workflow Scheduling 71
4.6 Taxonomy of Cloud Resource Scheduling 72
4.7 Existing Workflow Scheduling Algorithms 73
4.7.1 Best Effort Workflow Scheduling 73
4.7.2 Bi-objective Workflow Scheduling 74
4.7.3 Multi-objective Workflow Scheduling 75
4.8 Issues of Scheduling Workflow in Cloud 79
4.9 Conclusion 80
References 80

5 Workflow Modeling and Simulation Techniques 85
5.1 Introduction 85
5.2 Architecture of CloudSim 86
5.3 Layered Design and Implementation of CloudSim Framework 87
5.4 Experimental Results Using CloudSim 91
5.5 WorkflowSim 95

5.6 Architecture of WorkflowSim 95
5.7 Conclusion 101
References 101

6 Execution of Workflow Scheduling in Cloud Middleware 103
6.1 Introduction 103
6.2 Workflow Management System 105
6.3 Experimental Setup 106
6.4 General Steps for Submitting Workflow in Pegasus 107
6.4.1 Pegasus Monitoring and Measuring Service 114
6.5 Conclusion 117
References 117

7 Workflow Predictions Through Operational Analytics and Machine Learning 119
7.1 Introduction 119
7.2 Workflow Prediction 120
7.2.1 Challenges in Designing an APPS 121
7.2.2 Workflow Prediction Approaches 121
7.3 AM-Based Performance Prediction Systems 122
7.4 MLM-Based Performance Prediction Systems 125
7.5 Hybrid Performance Prediction Systems (HPPS) 127
7.5.1 Opportunities and Challenges for HPPS 127
7.6 Case Study 128
7.6.1 PANORAMA: An Approach to Performance Modeling and Diagnosis of Extreme-Scale Workflows 129
7.7 Conclusion 133
References 134

8 Workflow Integration and Orchestration, Opportunities and the Challenges 137
8.1 Introduction 137
8.2 Workflow Life Cycle 138
8.2.1 Workflow Creation 140
8.2.2 Workflow Mapping 141
8.2.3 Workflow Execution 141
8.2.4 Metadata and Provenance 142
8.3 Challenges and Opportunities 143
8.4 BioCloud: A Resource Provisioning Framework for Bioinformatics Applications in Multi-cloud Environments 145
8.4.1 BioCloud System Design 146
8.4.2 BioCloud in Action 150
8.5 Conclusion 153
References 153

9 Workload Consolidation Through Automated Workload Scheduling 157
9.1 Introduction 157
9.2 Workflow Scheduling 158
9.3 Cost-Based Scheduling Algorithm 159
9.3.1 Proportional Deadline Constrained Algorithm 159
9.3.2 Particle Swam Optimization (PSO) Algorithm 160
9.3.3 Hybrid Cloud Optimized Cost (HCOC) Schedule Algorithm 162
9.3.4 Ant Colony Optimization (ACO) Algorithm 163
9.3.5 Customer-Facilitated Cost-Based Scheduling in Cloud (CFSC) 164
9.3.6 Task Selection Priority and Deadline (TPD) Algorithm 164
9.4 Energy-Based Scheduling Algorithms 165
9.5 Automated Workload Consolidation 171
9.6 Conclusion 175
References 175

10 Automated Optimization Methods for Workflow Execution 177
10.1 Introduction 177
10.2 Workflow Optimization: Current State of the Art 178
10.3 Modified Workflow Life Cycle and Optimization Levels 180
10.3.1 Optimization Levels 181
10.4 Taverna Optimization Framework 183
10.5 Optimization Using Distributed Computing Infrastructure 186
10.5.1 Three-Tier Execution Architecture 186
10.5.2 Parallel Workflow Execution 187
10.5.3 Parallel Optimization Use Case 188
10.6 Optimization Techniques for Workflow Parameters 188
10.6.1 Genetic Algorithm 189
10.7 Parameter Optimization Plug-In 190
10.8 Validating Parameter Optimization Plug-In 191
10.8.1 Proteomics Workflow 192
10.8.2 Biomarker Identification Workflows 194
10.9 Conclusion 196
References 196

11 The Hybrid IT, the Characteristics and Capabilities 199
11.1 Introduction 199
11.2 Demystifying the Hybrid Cloud Paradigm 200
11.3 The Key Drivers for Hybrid Clouds 202
11.4 The Hybrid Cloud Challenges 204
11.5 Hybrid Cloud Management: The Use Cases and Requirements 208

11.6 The Growing Ecosystem of Hybrid Cloud Management Solutions 208
11.7 IBM Cloudmatrix-Based Multi-cloud Environments 210
11.8 The Key Drivers for Cloud Brokerage Solutions and Services 211
11.9 Benefits of Having a Cloud Broker 217
11.10 Conclusion 217
Appendix 217
Bibliography 221

Index 223

Chapter 1
Stepping into the Digital Intelligence Era

Abstract There are several noteworthy implications of the digitization movement. A myriad of digitation-enabling processes, platforms, products, patterns, and practices are hitting the market. There is a kind of convergence blurring the boundaries of physical and virtual worlds. Further on, all kinds of physical, mechanical, electrical, and electronics systems are adequately empowered to be instrumented and interconnected to exhibit intelligent behavior. Further on, all kinds of connected devices and digitized objects in our everyday environments are systematically integrated with remotely held and cloud-hosted applications, services, and data sources in order to be adaptive in their deeds, decisions, and deliveries.

The faster maturity and stability of edge technologies such as bar codes, application-specific chips, microcontrollers, stickers, labels, tags, smart dust, specks, sensors, actuators, LED lights, etc. speeds up the production of digitized objects, alternatively termed as smart or sentient materials. Every common, casual, and cheap item can be transitioned into digitized entities and elements in order to join in the mainstream computing. Next in line is that the explosion of instrumented and connected devices that are very generic and specific. That is, there are portable, wearable, wireless, mobile, implantable, hearable, and nomadic devices in plenty. Now with the emergence of microservices architecture (MSA), all kinds of enterprise-scale, operational, analytical, and transactional applications are being designed and developed using the MSA pattern. Thus scores of ground-level digitized elements in conjunction with cyber/virtual applications being run on cloud platforms and infrastructures strengthen the digitization era. This chapter is to tell all about the digitization-inspired possibilities and opportunities and how software-defined cloud centers are the best fit for digital applications.

In this chapter, we are to describe some of the impactful developments brewing in the IT space, how the tremendous amount of data getting produced and processed all over the world is to impact the IT and business domains, and how next-generation IT infrastructures are accordingly getting refactored, remedied, and readied for the impending big data-induced challenges, how likely the move of the big data analytics discipline toward fulfilling the digital universe requirements of extracting and extrapolating actionable insights for the knowledge-parched is, and finally for the establishment and sustenance of the dreamt smarter planet.

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_1

1.1 Introduction

One of the most visible and value-adding trends in IT is nonetheless the digitization aspect. All kinds of everyday items in our personal, professional, and social environments are being digitized systematically to be computational, communicative, sensitive, perceptive, and responsive. That is, all kinds of ordinary entities in our midst are instrumented differently to be extraordinary in their operations, outputs, and offerings. These days, due to the unprecedented maturity and stability of a host of path-breaking technologies such as miniaturization, integration, communication, computing, sensing, perception, middleware, analysis, actuation, and orchestration, everything has grasped the inherent power of interconnecting with one another in its vicinity as well as with remote objects via networks purposefully and on need basis to uninhibitedly share their distinct capabilities toward the goal of business automation, acceleration, and augmentation. Ultimately, everything will become smart, electronics goods will become smarter, and human beings will become the smartest in their deals, decisions, and deeds.

1.2 Elucidating Digitization Technologies

In this section, the most prevalent and pioneering trends and transitions in the IT landscape will be discussed. Especially the digitization technologies and techniques are given the sufficient thrust.

The Trend-Setting Technologies in the IT Space As widely reported, there are several delectable transitions in the IT landscape. The consequences are vast and varied: the incorporation of nimbler and next-generation features and functionalities into existing IT solutions, the grand opening of fresh possibilities and opportunities, and the eruption of altogether new IT products and solutions for the humanity. These have the intrinsic capabilities to bring forth numerous subtle and succinct transformations in business as well as people.

IT Consumerization The much-discoursed and deliberated Gartner report details the diversity of mobile devices (smartphones, tablets, wearables, drones, etc.) and their management. This trend is ultimately empowering people. The ubiquitous information access is made possible. Further on, the IT infrastructures are being tweaked accordingly in order to support this movement. There are some challenges for IT administrators in fulfilling the device explosion. That is, IT is steadily becoming an inescapable part of consumers directly and indirectly. And the need for robust and resilient mobile device management software solutions with the powerful emergence of “bring your own device (BYOD)” is being felt and is being insisted across. Another aspect is the emergence of next-generation mobile applications and services across a variety of business verticals. There is a myriad of mobile applications, maps and services development platforms, programming and markup languages, architectures and frameworks, tools, containers, and operating systems in the fast-moving mobile space.

IT Commoditization This is another cool trend penetrating in the IT industry. With the huge acceptance and adoption of cloud computing and big data analytics, the value of commodity IT is decidedly on the rise. The embedded intelligence inside IT hardware elements is being abstracted and centralized in hypervisor software solutions. Hardware systems are thus software enabled to be flexibly manipulated and maneuvered. With this transition, all the hardware resources in any data center become dumb and can be easily replaced, substituted, and composed for easily and quickly fulfilling different requirements and use cases. The IT affordability is thus realized along with a number of other advantages. That is, the future IT data centers and server farms are going to be stuffed with a number of commodity servers, storages, and network solutions. The utilization level is bound to go up substantially when the IT resources are commoditized.

IT Compartmentalization (Virtualization and Containerization) The “divide and conquer” has been the most versatile and rewarding mantra in the IT field. Abstraction is another powerful technique gaining a lot of ground. The widely used virtualization, which had laid a stimulating and sustainable foundation for the raging cloud idea, is actually hardware virtualization. Then the aspect of containerization being represented through the popular Docker containers is one step above the hardware virtualization. That is, containerization is at the operating system (OS)-level virtualization, which is incidentally lightweight. These two are clearly and cleverly leading to the cloud journey now.

IT Digitization and Distribution As explained in the beginning, digitization has been an ongoing and overwhelming process, and it has quickly generated and garnered a lot of market and mind shares. Digitally enabling everything around us induces a dazzling array of cascading and captivating effects in the form of cognitive and comprehensive transformations for businesses as well as people. With the growing maturity and affordability of edge technologies, every common thing in our personal, social, and professional environment is becoming digitized. Devices are being tactically empowered to be intelligent. Ordinary articles are becoming smart artifacts in order to significantly enhance the convenience, choice, and comfort levels of humans in their everyday lives and works. Therefore it is no exaggeration in stating that lately there have been a number of tactical as well as strategic advancements in the edge-technologies space. Infinitesimal and invisible tags, sensors, actuators, controllers, stickers, chips, codes, motes, specks, smart dust, and the like are being produced in plenty. Every single tangible item in our midst is being systematically digitized by internally as well as externally attaching these minuscule products onto them. This is for empowering them to be smart in their actions and reactions.

Similarly, the distribution aspect too gains more ground. Due to its significant advantages in crafting and sustaining a variety of business applications ensuring the hard-to-realize quality of service (QoS) attributes, there is a bevy of distribution-centric software architectures, frameworks, patterns, practices, and platforms for the Web, enterprise, embedded, analytical, and cloud applications and services.

1.2.1 Why Digitization?

Ultimately all kinds of perceptible objects in our everyday environments will be empowered to be self-, surroundings-, and situation-aware; remotely identifiable; readable; recognizable; addressable; and controllable. Such a profound empowerment will bring forth transformations for the total human society, especially in establishing and sustaining smarter environments, such as smarter homes, buildings, hospitals, classrooms, offices, and cities. Suppose, for instance, a disaster occurs. If everything in the disaster area is digitized, then it becomes possible to rapidly determine what exactly has happened, the intensity of the disaster, and the hidden risks of the affected environment. Any information extracted provides a way to plan and proceed insightfully, reveals the extent of the destruction, and conveys the correct situation of the people therein. The knowledge gained would enable the rescue and emergency team leaders to cognitively contemplate appropriate decisions and plunge into actions straightaway to rescue as much as possible, thereby minimizing damage and losses.

In short, digitization will enhance our decision-making capability in our personal as well as professional lives. Digitization also means that the ways we learn and teach are to change profoundly, energy usage will become knowledge-driven so that green goals can be met more smoothly and quickly, and the security and safety of people and properties will go up considerably. As digitization becomes pervasive, our living, relaxing, working, and other vital places will be filled up with a variety of electronics including environment monitoring sensors, actuators, monitors, controllers, processors, projectors, displays, cameras, computers, communicators, appliances, gateways, high-definition IP TVs, and the like. In addition, items such as furniture and packages will become empowered by attaching specially made electronics onto them. Whenever we walk into such kinds of empowered environments, the devices we carry and even our e-clothes will enter into collaboration mode and from wireless ad hoc networks with the objects in that environment. For example, if someone wants to print a document from their smartphone or tablet and they enter into a room where a printer is situated, the smartphone will automatically begin a conversation with the printer, check its competencies, and send the documents to be printed. The smartphone will then alert the owner.

Digitization will also provide enhanced care, comfort, choice, and convenience. Next-generation healthcare services will demand deeply connected solutions. For example, Ambient Assisted Living (AAL) is a new prospective application domain where lonely, aged, diseased, bedridden, and debilitated people living at home will receive are mote diagnosis, care, and management as medical doctors, nurses, and other caregivers remotely monitor patients' health parameters.

People can track the progress of their fitness routines. Taking decisions becomes an easy and timely affair with the prevalence of connected solutions that benefit knowledge workers immensely. All the secondary and peripheral needs will be accomplished in an unobtrusive manner people to nonchalantly focus on their primary activities. However, there are some areas of digitization that need attention, one being energy efficient. Green solutions and practices are being insisted upon everywhere these days, and IT are one of the principal culprits in wasting a lot of energy due to the pervasiveness of IT servers and connected devices. Data centers consume a lot of electricity, so green IT is a hot subject for study and research across the globe. Another area of interest is remote monitoring, management, and enhancement of the empowered devices. With the number of devices in our everyday environments growing at an unprecedented scale, their real-time administration, configuration, activation, monitoring, management, and repair (if any problem arises) can be eased considerably with effective remote connection and correction competency.

Extreme Connectivity The connectivity capability has risen dramatically and become deeper and extreme. The kinds of network topologies are consistently expanding and empowering their participants and constituents to be highly productive. There are unified, ambient, and autonomic communication technologies from research organizations and labs drawing the attention of executives and decision-makers. All kinds of systems, sensors, actuators, and other devices are empowered to form ad hoc networks for accomplishing specialized tasks in a simpler manner. There are a variety of network and connectivity solutions in the form of load balancers, switches, routers, gateways, proxies, firewalls, etc. for providing higher performance; network solutions are being embedded in appliances (software as well as hardware) mode.

Device middleware or Device Service Bus (DSB) is the latest buzzword enabling a seamless and spontaneous connectivity and integration between disparate and distributed devices. That is, device-to-device (in other words, machine-to-machine (M2M)) communication is the talk of the town. The interconnectivity-facilitated interactions among diverse categories of devices precisely portend a litany of supple, smart, and sophisticated applications for people. Software-defined networking (SDN) is the latest technological trend captivating professionals to have a renewed focus on this emerging yet compelling concept. With clouds being strengthened as the core, converged, and central IT infrastructure, device-to-cloud connections are fast-materializing. This local as well as remote connectivity empowers ordinary articles to become extraordinary objects by being distinctively communicative, collaborative, and cognitive.

Service Enablement Every technology pushes for its adoption invariably. The Internet computing has forced for the beneficial Web enablement, which is the essence behind the proliferation of Web-based applications. Now, with the pervasiveness of sleek, handy, and multifaceted mobiles, every enterprise and Web applications are being mobile-enabled. That is, any kind of local and remote applications are being accessed through mobiles on the move, thus fulfilling real-time interactions and decision-making economically. With the overwhelming approval of the service idea, every application is service-enabled. That is, we often read, hear, and feel service-oriented systems. The majority of next-generation enterprise-scale, mission-critical, process-centric, and multi-purpose applications are being assembled out of multiple discrete and complex services.

Not only applications but physical devices at the ground level are being seriously service-enabled in order to uninhibitedly join in the mainstream computing tasks and contribute to the intended success. That is, devices, individually and collectively, could become service providers or publishers, brokers and boosters, and consumers. The prevailing and pulsating idea is that any service-enabled device in a physical environment could interoperate with others in the vicinity as well as with remote devices and applications. Services could abstract and expose only specific capabilities of devices through service interfaces while service implementations are hidden from user agents. Such kinds of smart separations enable any requesting device to see only the capabilities of target devices and then connect, access, and leverage those capabilities to achieve business or people services. The service enablement completely eliminates all dependencies and deficiencies so that devices could interact with one another flawlessly and flexibly.

1.3 The Internet of Things (IoT)/Internet of Everything (IoE)

Originally, the Internet was the network of networked computers. Then, with the heightened ubiquity and utility of wireless and wired devices, the scope, size, and structure of the Internet has changed to what it is now, making the Internet of Devices (IoD) concept a mainstream reality. With the service paradigm being positioned as the most optimal, rational, and practical way of building enterprise-class applications, a gamut of services (business and IT) are being built by many, deployed in worldwide Web and application servers and delivered to everyone via an increasing array of input/output devices over networks. The increased accessibility and audibilityof services have propelled interested software architects, engineers, and application developers to realize modular, scalable, and secured software applications by choosing and composing appropriate services from those service repositories quickly. Thus, the Internet of Services (IoS) idea is fast-growing. Another interesting phenomenon getting the attention of press these days is the Internet of Energy. That is, our personal, as well as professional, devices get their energy through their interconnectivity. Figure 1.1 clearly illustrates how different things are linked with one another in order to conceive, concretize, and deliver futuristic services for the mankind (Distributed Data Mining and Big Data, a Vision paper by Intel, 2012).

As digitization gains more accolades and success, all sorts of everyday objects are being connected with one another as well as with scores of remote applications in cloud environments. That is, everything is becoming a data supplier for the next-generation applications, thereby becoming an indispensable ingredient individually as well as collectively in consciously conceptualizing and concretizing smarter applications. There are several promising implementation technologies, standards, platforms, and tools enabling the realization of the IoT vision. The probable outputs of the IoT field is a cornucopia of smarter environments such as smarter offices,

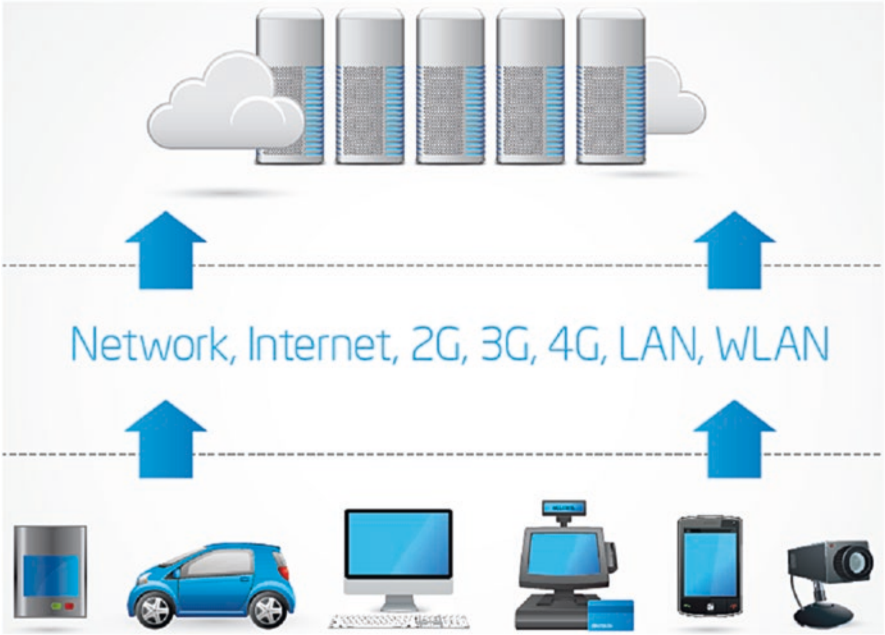


Fig. 1.1 The extreme connectivity among physical devices with virtual applications

homes, hospitals, retail, energy, government, cities, etc. Cyber-physical systems (CPS), ambient intelligence (AmI), and ubiquitous computing (UC) are some of the related concepts encompassing the ideals of IoT.

In the upcoming era, unobtrusive computers, communicators, and sensors will be facilitating decision-making in a smart way. Computers in different sizes, looks, capabilities, and interfaces will be fitted, glued, implanted, and inserted everywhere to be coordinative, calculative, and coherent. The interpretation and involvement of humans in operationalizing these smarter and sentient objects are almost zero. With autonomic IT infrastructures, more intensive automation is bound to happen. The devices will also be handling all kinds of everyday needs, with humanized robots extensively used in order to fulfill our daily physical chores. With the emergence of specific devices for different environments, there will similarly be hordes of services and applications coming available for making the devices smarter that will, in turn, make our lives more productive.

On summarizing, the Internet is expanding into enterprise assets and consumer items such as cars and televisions. Gartner identifies four basic usage models that are emerging:

- Manage
- Monetize
- Operate
- Extend

These can be applied to people, things, information, and places. That is, the Internet of Everything is all set to flourish unflinchingly. Let me summarize and categorize the delectable advancements in the ICT discipline into three.

Tending Toward the Trillions of Digitized Elements/Smart Objects/Sentient Materials The surging popularity and pervasiveness of a litany of digitization and edge technologies such as sensors, actuators, and other implantables; application-specific system-on-a-chip (SoC); microcontrollers; miniaturized RFID tags; easy-to-handle barcodes, stickers, and labels; nanoscale specks and particles, illuminating LED lights, etc. come handy in readily enabling every kind of common, casual, and cheap things in our everyday environments to join in the mainstream computing. All kinds of ordinary and tangible things in our midst get succulently transitioned into purposefully contributing and participating articles to accomplish extraordinary tasks. This tactically as well as strategically beneficial transformation is performed through the well-intended and well-designed application of disposable and disappearing yet indispensable digitization technologies. In short, every concrete thing gets systematically connected with one another as well as the Web. This phenomenon is being aptly termed as the Internet of Things (IoT).

Further on, every kind of physical, mechanical, and electrical system in the ground level get hooked to various software applications and data sources at the faraway as well as nearby cyber/virtual environments. Resultantly, the emerging domain of cyber-physical systems (CPS) is gaining immense attention lately:

1. *Ticking toward the billions of connected devices* – A myriad of electronics, machines, instruments, wares, equipment, drones, pumps, wearables, hearables, robots, smartphones, and other devices across industry verticals are intrinsically instrumented at the design and manufacturing stages in order to embed the connectivity capability. These devices are also being integrated with software services and data sources at cloud environments to be enabled accordingly.
2. *Envisioning millions of software services* – With the accelerated adoption of microservices architecture (MSA), enterprise-scale applications are being expressed and exposed as a dynamic collection of fine-grained, loosely-coupled, network-accessible, publicly discoverable, API-enabled, composable, and lightweight services. This arrangement is all set to lay down a stimulating and sustainable foundation for producing next-generation software applications and services. The emergence of the scintillating concepts such as Docker containers, IT agility, and DevOps in conjunction with MSA clearly foretells that the days of software engineering is bound to flourish bigger and better. Even hardware resources and assets are being software-defined in order to incorporate the much-needed flexibility, maneuverability, and extensibility.

In short, every tangible thing becomes smart, every device becomes smarter, and every human being tends to become the smartest.

The disruptions and transformations brought in by the above-articulated advancements are really mesmerizing. The IT has touched every small or big entity decisively in order to produce context-aware, service-oriented, event-driven, knowledge-filled, people-centric, and cloud-hosted applications. Data-driven insights and insight-driven enterprises are indisputably the new normal.

Infrastructure Optimization The entire IT stack has been going for the makeover periodically. Especially on the infrastructure front due to the closed, inflexible, and monolithic nature of conventional infrastructure, there are concerted efforts being undertaken by many in order to untangle them into modular, open, extensible, converged, and programmable infrastructures. Another worrying factor is the underutilization of expensive IT infrastructures (servers, storages, and networking solutions). With IT becoming ubiquitous for automating most of the manual tasks in different verticals, the problem of IT sprawl is to go up, and they are mostly underutilized and sometimes even unutilized for a long time. Having understood these prickling issues pertaining to IT infrastructures, the concerned have plunged into unveiling versatile and venerable measures for enhanced utilization and for infrastructure optimization. Infrastructure rationalization and simplification are related activities. That is, next-generation IT infrastructures are being realized through consolidation, centralization, federation, convergence, virtualization, automation, and sharing. To bring in more flexibility, software-defined infrastructures are being prescribed these days.

With the faster spread of big data analytics platforms and applications, commodity hardware is being insisted on accomplishing data and process-intensive big data analytics quickly and cheaply. That is, we need low-priced infrastructures with supercomputing capability and virtually infinite storage. The answer is that all kinds of underutilized servers are collected and clustered together to form a dynamic and huge pool of server machines to efficiently tackle the increasing and intermittent needs of computation. Precisely speaking, clouds are the new-generation infrastructures that fully comply with these expectations elegantly and economically. The cloud technology, though not a new one, represents a cool and compact convergence of several proven technologies to create a spellbound impact on both business and IT in realizing the dream of virtual IT that in turn blurs the distinction between the cyber and the physical worlds. This is the reason for the exponential growth being attained by the cloud paradigm. That is, the tried and tested technique of "divide and conquer" in software engineering is steadily percolating to hardware engineering. Decomposition of physical machines into a collection of sizable and manageable virtual machines and composition of these virtual machines based on the computing requirement is the essence of cloud computing.

Finally, software-defined cloud centers will see the light soon with the faster maturity and stability of competent technologies toward that goal. There is still some critical inflexibility, incompatibility, and tighter dependency issues among various components in cloud-enabled data centers, thus full-fledged optimization and automation are not yet possible with the current setup. To attain the originally envisaged goals, researchers are proposing to incorporate software wherever needed in order to bring in the desired separations so that a significantly higher utilization is possible. When the utilization goes up, the cost is bound to come down. In short, the target of infrastructure programmability can be met with the embedding of resilient software so that the infrastructure manageability, serviceability, and sustainability tasks become easier, economical, and quicker.

1.4 Real-Time, Predictive, and Prescriptive Analytics

As we all know, the big data paradigm is opening up a fresh set of opportunities for businesses. As data explosion would occur according to the forecasts of leading market research and analyst reports, the key challenge in front of businesses is how efficiently and rapidly to capture, process, analyze, and extract tactical, operational, and strategic insights in time to act upon swiftly with all the confidence and clarity. In the recent past, there were experiments on in-memory computing. For a faster generation of insights out of a large amount of multi-structured data, the new entrants such as in-memory and in-database analytics are highly reviewed and recommended. The new mechanism insists on putting all incoming data in memory instead of storing it in local or remote databases so that the major barrier of data latency gets eliminated. There are a variety of big data analytics applications in the market and they implement this new technique in order to facilitate real-time data analytics. Timeliness is an important factor for information to be beneficially leveraged. The appliances are in general high-performing, thus guaranteeing higher throughput in all they do. Here too, considering the need for real-time emission of insights, several product vendors have taken the route of software as well as hardware appliances for substantially accelerating the speed with which the next-generation big data analytics get accomplished.

In the business intelligence (BI) industry, apart from realizing real-time insights, analytical processes and platforms are being tuned to bring forth insights that invariably predict something to happen for businesses in the near future. Therefore, executives and other serious stakeholders proactively and preemptively can formulate well-defined schemes and action plans, fresh policies, new product offerings, premium services, and viable and value-added solutions based on the inputs. Prescriptive analytics, on the other hand, is to assist business executives inprescribing and formulating competent and comprehensive schemes and solutions based on the predicted trends and transitions.

IBM has introduced a new computing paradigm "stream computing" in order to capture streaming and event data on the fly and to come out with usable and reusable patterns, hidden associations, tips, alerts and notifications, impending opportunities, threats, etc. in time for executives and decision-makers to contemplate appropriate countermeasures (James Kobielus [2]).

The Recent Happenings in the IT Space

- *Extended Device Ecosystem* – Trendy and handy, slim and sleek mobile; wearable, implantable, and portable; and energy-aware devices (instrumented, interconnected, and intelligent devices)
- *Sentient and Smart Materials* – Attaching scores of edge technologies (invisible, calm, infinitesimal, and disposable sensors and actuators, stickers, tags, labels, motes, dots, specks, etc.) on ordinary objects to exhibit extraordinary capabilities

(continued)

- *Extreme and Deeper Connectivity* – Standards, technologies, platforms, and appliances for device-to-device, device-to-cloud, cloud-to-cloud, and on-premise to off-premise interactions
- *Infrastructure Optimization* – Programmable, consolidated, converged, adaptive, automated, shared, QoS-enabling, green, and lean infrastructures
- *Unified Platform and Middleware Solutions* – Intermediation, aggregation, dissemination, arbitration, enrichment, collaboration, delivery, management, governance, brokering, identity, and security
- *New-Generation Databases* – SQL, NoSQL, NewSQL, and hybrid databases for the big data world
- *Real-time, Predictive, and Prescriptive Analytics* – Big data analytics, in-memory computing, etc.
- *Process Innovation and Architecture Assimilation* – SOA, EDA, SCA, MDA, ROA, WOA, etc.
- *A Bevy of Pioneering Technologies* – Virtualization, miniaturization, integration, composition, sensing, vision, perception, mobility, knowledge engineering, visualization, etc.
- *Next-Generation Applications* – Social, mobile, cloud, enterprise, Web, analytical, and embedded application categories

The Big Picture With the cloud space growing fast as the next-generation environment for application design, development, deployment, integration, management, and delivery as a service, the integration requirement too has grown deeper and broader as pictorially illustrated in Fig. 1.2.

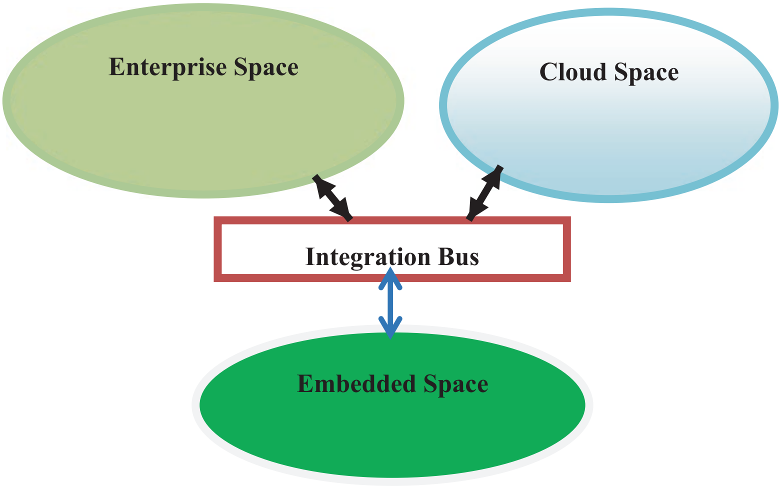


Fig. 1.2 The connected and integrated world

All kinds of physical entities at the ground level will have purpose-specific interactions with services and data hosted on the enterprise as well as cloud servers and storages to enable scores of real-time and real-world applications for the society. This extended and enhanced integration would lead to data deluges that have to be accurately and appropriately subjected to a variety of checks to promptly derive actionable insights that in turn enable institutions, innovators, and individuals to be smarter and speedier in their obligations and offerings. Newer environments such as smarter cities, governments, retail, healthcare, energy, manufacturing, supply chain, offices, and homes will flourish. Cloud, being the smartest IT technology, is inherently capable of meeting up with all kinds of infrastructural requirements fully and firmly.

1.5 Envisioning the Digital Universe

The digitization process has gripped the whole world today as never before, and its impacts and initiatives are being widely talked about. With an increasing variety of input and output devices and newer data sources, the realm of data generation has gone up remarkably. It is forecasted that there will be billions of everyday devices getting connected, capable of generating an enormous amount of data assets which need to be processed. It is clear that the envisaged digital world is to result in a huge amount of bankable data. This growing data richness, diversity, value, and reach decisively gripped the business organizations and governments first. Thus, there is a fast spreading of newer terminologies such as digital enterprises and economies. Now it is fascinating the whole world, and this new world order hastellingly woken up worldwide professionals and professors to formulate and firm up flexible, futuristic strategies, policies, practices, technologies, tools, platforms, and infrastructures to tackle this colossal yet cognitive challenge head on. Also, IT product vendors are releasing refined and resilient storage appliances, newer types of databases, distributed file systems, data warehouses, etc. to stock up for the growing volume of business, personal, machine, people, and online data and to enable specific types of data processing, mining, slicing, and analyzing the data getting collected and processed. This pivotal phenomenon has become a clear reason for envisioning the digital universe.

There will be hitherto unforeseen applications in the digital universe in which all kinds of data producers, middleware, consumers, storages, analytical systems, virtualization, and visualization tools and software applications will be seamlessly and spontaneously connected with one another. Especially there is a series of renowned and radical transformations in the sensor space. Nanotechnology and other miniaturization technologies have brought in legendary changes in sensor design. The nanosensors can be used to detect vibrations, motion, sound, color, light, humidity, chemical composition, and many other characteristics of their deployed environments. These sensors can revolutionize the search for new oil reservoirs, structural integrity for buildings and bridges, merchandise tracking and authentication, food and water safety, energy use and optimization, healthcare monitoring and cost savings, and climate and environmental monitoring. The point to be noted here is the volume of real-time data being emitted by the army of sensors and actuators.

The steady growth of sensor networks increases the need for 1 million times more storage and processing power by 2020. It is projected that there will be one trillion sensors by 2030 and every single person will be assisted by approximately 150 sensors in this planet. Cisco has predicted that there will be 50 billion connected devices in 2020, and hence the days of the Internet of Everything (IoE) are not too far off. All these scary statistics convey one thing, which is that IT applications, services, platforms, and infrastructures need to be substantially and smartly invigorated to meet up all sorts of business and peoples' needs in the ensuing era of deepened digitization.

Precisely speaking, the data volume is to be humongous as the digitization is growing deep and wide. The resulting digitization-induced digital universe will, therefore, be at war with the amount of data being collected and analyzed. The data complexity through the data heterogeneity and multiplicity will be a real challenge for enterprise IT teams. Therefore big data is being positioned and projected as the right computing model to effectively tackle the data revolution challenges of the ensuing digital universe.

1.6 Describing the Digitization-Driven Big Data World

Big Data Analytics (BDA) The big data paradigm has become a big topic across nearly every business domain. IDC defines big data computing as a set of new-generation technologies and architectures, designed to economically extract value from very large volumes of a wide variety of data by enabling high-velocity capture, discovery, and/or analysis. There are three core components in big data: the data itself, the analytics of the data captured and consolidated, and the articulation of insights oozing out of data analytics. There are robust products and services that can be wrapped around one or all of these big data elements. Thus there is a direct connectivity and correlation between the digital universe and the big data idea sweeping the entire business scene. The vast majority of new data being generated as a result of digitization is unstructured or semi-structured. This means there is a need arising to somehow characterize or tag such kinds of multi-structured big data to be useful and usable. This empowerment through additional characterization or tagging results in metadata, which is one of the fastest-growing subsegments of the digital universe through metadata itself, is a minuscule part of the digital universe. IDC believes that by 2020, a third of the data in the digital universe (more than 13,000 exabytes) will have big data value, only if it is tagged and analyzed. There will be routine, repetitive, redundant data and hence not all data is necessarily useful for big data analytics. However, there are some specific data types that are princely ripe for big analysis such as:

Surveillance Footage Generic metadata (date, time, location, etc.) is automatically attached to video files. However, as IP cameras continue to proliferate, there is a greater opportunity to embed more intelligence into the camera on the edges so that footage can be captured, analyzed, and tagged in real time. This type of tagging can expedite crime investigations for security insights, enhance retail analytics for consumer traffic patterns, and of course improve military intelligence as videos from drones across multiple geographies are compared for pattern correlations, crowd emergence, and response or measuring the effectiveness of counterinsurgency.

Embedded and Medical Devices In the future, sensors of all types including those that may be implanted into the body will capture vital and non-vital biometrics, track medicine effectiveness, correlate bodily activity with health, monitor potential outbreaks of viruses, etc. all in real time, thereby realizing automated healthcare with prediction and precaution.

Entertainment and Social Media Trends based on crowds or massive groups of individuals can be a great source of big data to help bring to market the "next big thing," help pick winners and losers in the stock market, and even predict the outcome of elections all based on information users freely publish through social outlets.

Consumer Images We say a lot about ourselves when we post pictures of ourselves or our families or friends. A picture used to be worth a thousand words, but the advent of big data has introduced a significant multiplier. The key will be the introduction of sophisticated tagging algorithms that can analyze images either in real time when pictures are taken or uploaded or en masse after they are aggregated from various Websites.

Data Empowers Consumers Besides organizations, digital data helps individuals to navigate the maze of modern life. As life becomes increasingly complex and intertwined, digital data will simplify the tasks of decision-making and actuation. The growing uncertainty in the world economy over the last few years has shifted many risk management responsibilities from institutions to individuals. In addition to this increase in personal responsibility, other pertinent issues such as life insurance, healthcare, retirement, etc. are growing evermore intricate, increasing the number of difficult decisions we all make very frequently. The data-driven insights come handy in difficult situations for consumers to wriggle out. Digital data, hence, is the foundation and fountain of the knowledge society.

Power Shifts to the Data-Driven Consumers Data is an asset for all. Organizations are sagacious and successful in promptly bringing out the premium and people-centric offerings by extracting operational and strategically sound intelligence out of accumulated business, market, social, and people data. There is a gamut of advancements in data analytics in the form of unified platforms and optimized algorithms for efficient data analysis, portables and hearables, etc. There are plenty of data virtualization and visualization technologies. These give customers enough confidence and ever-greater access to pricing information, service records, and specifics on business behavior and performance. With the new-generation data analytics being performed easily and economically in cloud platforms and transmitted to smartphones, the success of any enterprise or endeavor solely rests with knowledge-empowered consumers.

Consumers Delegate Tasks to Digital Concierges We have been using a myriad of digital assistants (tablets, smartphones, wearables, etc.) for a variety of purposes in our daily life. These electronics are of great help, and crafting applications and services for these specific as well as generic devices empower them to be more right and relevant for us. Data-driven smart applications will enable these new-generation digital concierges to be expertly tuned to help us in many things in our daily life.

Big data is driving a revolution in machine learning and automation. This will create a wealth of new smart applications and devices that can anticipate our needs precisely and perfectly. In addition to responding to requests, these smart applications will proactively offer information and advice based on detailed knowledge of our situation, interests, and opinions. This convergence of data and automation will simultaneously drive a rise of user-friendly analytic tools that help to make sense of the information and create new levels of ease and empowerment for everything from data entry to decision-making. Our tools will become our data interpreters, business advisors, and life coaches, making us smarter and more fluent in all subjects of life.

Data Fosters Community Due to the growing array of extra facilities, opportunities, and luxuries being made available and accessible in modernized cities, there is a consistent migration to urban areas and metros from villages. This trend has displaced people from their roots and there is a huge disconnect between people in new locations also. Now with the development and deployment of services (online location-based services, local search, community-specific services, and new data-driven discovery applications) based on the growing size of social, professional, and people data, people can quickly form digital communities virtually in order to explore, find, share, link, and collaborate with others. The popular social networking sites enable people to meet and interact with one another purposefully. The government uses data and analytics to establish citizen-centric services, improve public safety, and reduce crime. Medical practitioners use it to diagnose better and treat diseases effectively. Individuals are tapping on online data and tools for help with everything from planning their career to retirement, to choose everyday service providers, to pick up places to live, to find the quickest way to get to work, and so on. Data, services, and connectivity are the three prime ingredients in establishing and sustaining rewarding relationships among diverse and distributed people groups.

Data Empowers Businesses to Be Smart Big data is changing the way companies conduct businesses. Starting with streamlining operations, increasing efficiencies to boost the productivity, improving decision-making, and bringing forth premium services to market are some of the serious turnarounds due to big data concepts. It is all “more with less.” A lot of cost savings are being achieved by leveraging big data technologies smartly, and this, in turn, enables businesses to incorporate more competencies and capabilities.

Big data is also being used to better target customers, personalize goods and services, and build stronger relationships with customers, suppliers, and employees. Business will see intelligent devices, machines, and robots taking over many repetitive, mundane, difficult, and dangerous activities. Monitoring and providing real-time information about assets, operations, and employees and customers, these smart machines will extend and augment human capabilities. Computing power will increase as costs decrease. Sensors will monitor, forecast, and report on environments; smart machines will develop, share, and refine new data into a knowledge based on their repetitive tasks. Real-time, dynamic, analytics-based insights will help businesses provide unique services to their customers on the fly. Both sources will transmit these rich streams of data to cloud environments so that all kinds of implantable, wearable, portable, fixed, nomadic, and any input/output devices can provide timely information and insights to their users unobtrusively. There is a gamut of improvisations such as the machine learning discipline solidifying an ingenious foundation for smart devices. Scores of data interpretation engines, expert systems, and analytical applications go a long way in substantially augmenting and assisting humans in their decision-making tasks.

Big Data Brings in Big Opportunities The big data and cloud paradigms have collectively sparked a stream of opportunities for both start-ups, and existing small businesses find innovative ways to harness the power of the growing streams of digital data. As the digital economy and enterprise mature, there can be more powerful and pioneering products, solutions, and services.

1.7 The Cloud Infrastructures for the Digitization Era

In the beginning, we had written about three kinds of data being produced. The processing is mainly two types: batch and online (real-time) processing. As far as the speed with which data needs to be captured and processed is concerned, there are both low-latency as well as high-latency data. Therefore, the core role of stream computing (introduced by IBM) is to power extremely low-latency data, but it should not rely on high-volume storage to do its job. By contrast, the conventional big data platforms involve a massively parallel processing architecture comprising enterprise data warehouses (EDW), Hadoop framework, and other analytics databases. This setup usually requires high-volume storage that can have a considerable physical footprint within the data center. On the other hand, a stream computing architecture uses smaller servers distributed across many data centers. Therefore there is a need for blending and balancing of stream computing with the traditional one. It is all about choosing a big data fabric that elegantly fits for the purpose on hand. The big data analytics platform has to have specialized “data persistence” architectures for both short-latency persistence (caching) of in-motion data (stream computing) and long-latency persistence (storage) of at-rest data. Stream computing is for extracting actionable insights in time out of streaming data. This computing model prescribes an optimal architecture for real-time analysis for those data in flight.

Big Data Analytics Infrastructures And as IT moves to the strategic center of business, CXOs at organizations of all sizes turn to product vendors and service providers to help them extract more real value from their data assets, business processes, and other key investments. IT is being primed for eliminating all kinds of existing business and IT inefficiencies, slippages, wastages etc. Nearly 70% of the total IT budget is being spent on IT operations and maintenance alone. Two-thirds of companies go over schedule on their project deployments. Hence, this is the prime time to move into smarter computing through the systematic elimination of IT complexities and all the inflicted barriers to innovation. Thus there is a business need for a new category of systems. Many prescribe different characteristics for next-generation IT infrastructures. The future IT infrastructures need to be open, modular, dynamic, converged, instant-on, expertly integrated, shared, software-defined, virtualized, etc.

Infrastructure Clouds: The Foundation for Big Data Analytics Businesses currently face a deluge of data, and business leaders need a smart way of capturing and understanding information rapidly. Infrastructure clouds enable big data analytics in two ways: storage and analysis. With data flowing in from a wide range of sources over a variety of networks, it is imperative that IT can store and make the data accessible to the business. Infrastructure clouds also enable enterprises to take the full advantages of big data by providing high-performing, scalable, and agile storage. But the real value comes from analyzing all of the data made available. The lure of breakthrough insights has led many lines of business to set up their own server and storage infrastructure, networking solutions, analytics platforms, databases, and applications. Big data appliances are also gaining market and mind shares to have a single and simplified set up for big data analytics.

However, they are often only analyzing narrow slivers of the full datasets available. Without a centralized point of aggregation and integration, data is collected in a fragmented way, resulting in limited or partial insights. Considering the data- and process-intensive nature of big data storage and analysis, cloud computing, storage, and network infrastructures are the best course of action. Private, public, and hybrid clouds are the smartest way of proceeding with big data analytics. Also, social data are being transmitted over the public and open Internet; public clouds seem to be a good bet for some specific big data analytical workloads. There are WAN optimization technologies strengthening the case for public clouds for effective and efficient big data analysis. Succinctly speaking, cloud environments with all the latest advancements in the form of software-defined networking, storage and compute infrastructures, cloud federation, etc. are the future of fit-for-purpose big data analysis. State-of-the-art cloud centers are right for a cornucopia of next-generation big data applications and services.

IBM has come out with expert integrated systems that ingeniously eliminate all kinds of inflexibilities and inefficiencies. Cloud services and applications need to be scalable and the underlying IT infrastructures need to be elastic. The business success squarely and solely depends on IT agility, affordability, and adaptability. Hence, the attention has turned toward the new smarter computing model, in which IT infrastructure is more simple, efficient, and flexible.

There are three aspects as far as smarter computing is concerned. The first one is to tune IT systems using the flexibility of general-purpose systems to optimize the systems for the specific business environment. The second one is to take advantage of the simplicity of appliances, and the final one is to leverage the elasticity of cloud infrastructures. The question is how can organizations get the best of all these options in one system? Expert integrated systems are therefore much more than a static stack of pre-integrated components: a server here, some database software there, serving a fixed application at the top, etc. Instead, these expert integrated systems are based on "patterns of expertise," which can dramatically improve the responsiveness of the business. Patterns of expertise automatically balance, manage, and optimize the elements necessary from the underlying hardware resources up through the middleware and software to deliver and manage today's modern business processes, services, and applications. Thus, as far as infrastructures are concerned, expertly integrated systems are the most sought-after in the evolving field of big data analytics.

There are other players in the market producing adaptive infrastructures for making big data analytics easy. For example, HP talks about converged cloud infrastructures. At the heart of HP converged infrastructure (Fig. 1.3) is the ultimate end state of having any workload, anywhere, anytime. This is achieved through a systematic approach that brings all the server, storage, and networking resources together into a common pool. This approach also brings together management tools, policies, and processes so that resources and applications are managed in a holistic, integrated manner. And it brings together security and power and cooling management capabilities so systems and facilities work together to extend the life of the data center.

This all starts by freeing assets trapped in operations or by deploying a new converged infrastructure that establishes a service-oriented IT organization that better aligns IT with the wide variety of fluctuating business demands. This is exactly what the converged infrastructure does. It integrates and optimizes technologies into pools of interoperable resources so they can deliver operational flexibility. And as a business grows, a converged infrastructure provides the foundation for an instant-on enterprise. This type of organization shortens the time needed to provision infrastructure for new and existing enterprise services to drive competitive and service advantages—allowing the business to interact with customers, employees, and partners more quickly and with increased personalization.

All kinds of infrastructure resources are being pooled and shared across, thereby increasing their utilization levels significantly and saving a lot of IT budget. For high-end applications such as Web 2.0 social sites, big data analytics, machine-to-machine (M2M) services, and high-performance applications such as genome research, climate modeling, drug discovery, energy exploration, weather forecasting, financial services, new materials design, etc., shared infrastructure is being recommended.

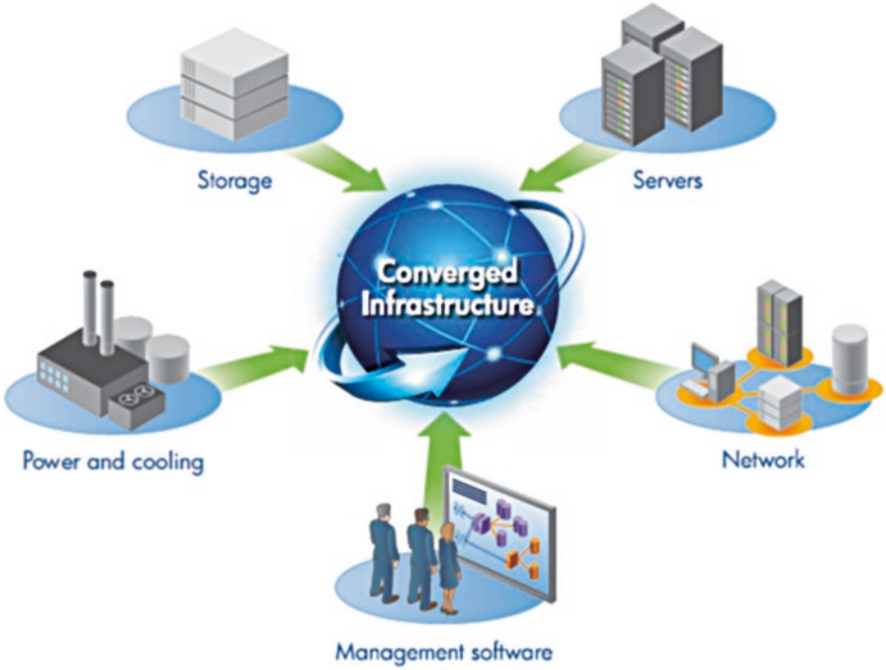


Fig. 1.3 The concept behind the converged infrastructure

1.8 Integrated Platform for Big Data Analytics (Dr. Barry Devlin [1])

Previously we have talked about versatile infrastructures for big data analytics. In this section, we are insisting on the need for integrated platforms for compact big data analysis. An integrated platform (Fig. 1.4) has to have all kinds of compatible and optimized technologies, platforms, and other ingredients to adaptively support varying business requirements.

The first is central core business data, the consistent, quality-assured data found in EDW and MDM systems. Traditional relational databases, such as IBM DB2, are the base technology. Application-specific reporting and decision support data often stored in EDWs today are excluded. Core reporting and analytic data cover the latter data types. In terms of technology, this type is ideally a relational database. Data warehouse platforms, such as IBM InfoSphere Warehouse, IBM Smart Analytics System, and the new IBM PureData System for Operational Analytics, play a strong role here. Business needs requiring higher query performance may demand an analytical database system built on massively parallel processing (MPP) columnar databases or other specialized technologies, such as the new IBM PureData System for Analytics (powered by Netezza technology).

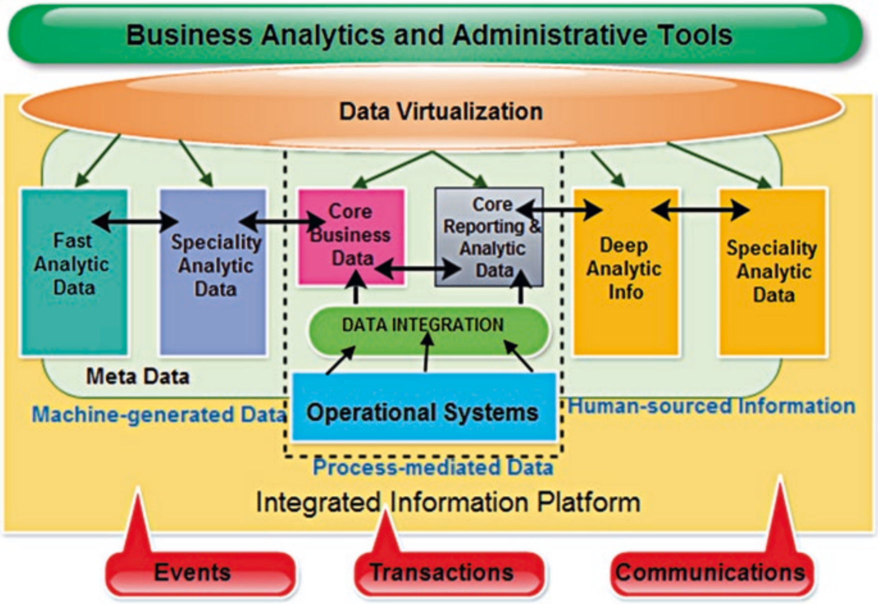


Fig. 1.4 The reference architecture for integrated data analytics platform

Deep analytic information requires highly flexible, large-scale processing such as the statistical analysis and text mining often performed in the Hadoop environment. Fast analytic data requires such high-speed analytic processing that it must be done on data in-flight, such as with IBM InfoSphere Streams, for example. This data is often generated from multiple sources that need to be continuously analyzed and aggregated with near-zero latency for real-time alerting and decision-making.

At the intersection of speed and flexibility, we have specialty analytic data, using specialized processing such as NoSQL, XML, graph, and other databases and data stores. Metadata, shown conceptually as a backdrop to all types of information, is central to this new architecture to define information context and to enable proper governance. In the process-mediated and machine-generated domains, metadata is explicitly stored separately; in the human-sourced domain, it is more likely to be implicit in the information itself. This demands new approaches to modeling, discovering, and visualizing both internal and external sources of data and their interrelationshipswithin the platform.

Transitioning from Traditional BI to Big Data BI (How to Enhance Traditional BI Architecture to Leverage Big Data, a White paper by [3])–Business intelligence (BI) has been a key requirement for every aspiring enterprise across the globe. There are consultants, empowered service organizations, and product vendors collaboratively strategizing and actuating to establish the BI competency for worldwide businesses (small, medium, and large) as this empowerment would innately help to plan and execute appropriate actions for business optimization and transformations. The BI horizon has now increased sharply with distributed and diverse data sources. The changes have propelled industry professionals, research labs, and academicians to bring in required technologies, tools, techniques, and tips. The traditional BI architecture (Fig. 1.5) is as follows:

The big data-inspired BI architecture is given below. There are additional modules in this architecture as big data analytics typically involves data collection, virtualization, pre-processing, information storage, and knowledge discovery and articulation activities (Fig. 1.6).

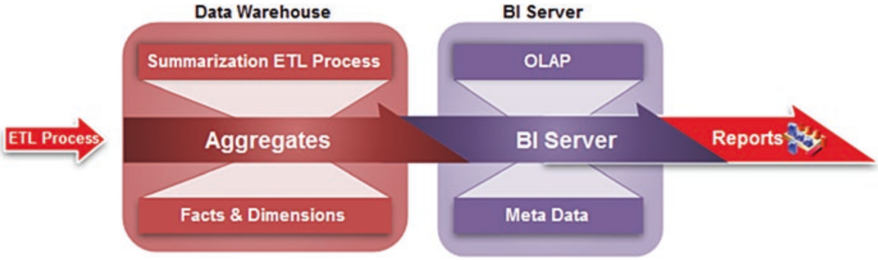


Fig. 1.5 The traditional business intelligence (BI) architecture

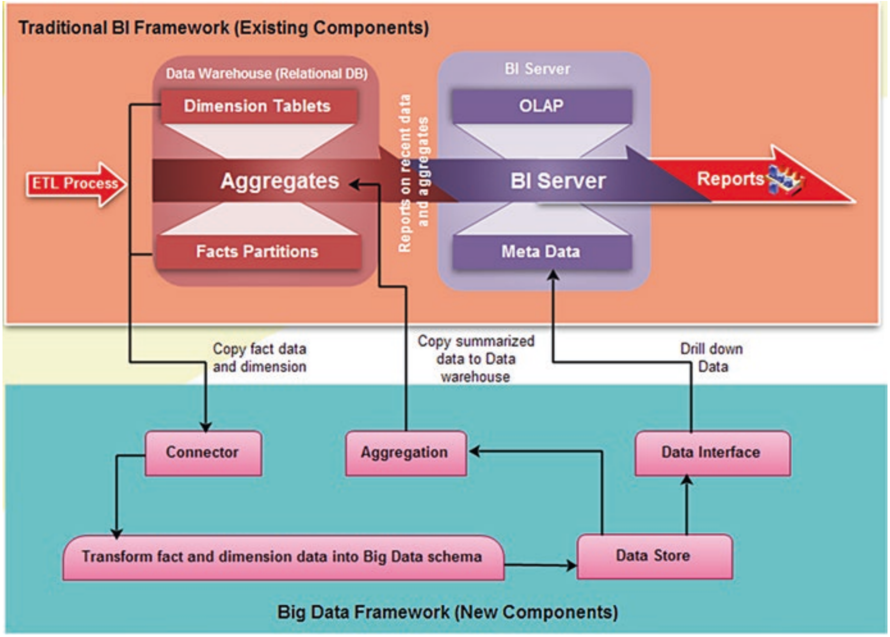

Traditional BI Framework (Existing Components)
Data Warehouse (Relational DB)
BI Server
Dimension Tablets
OLAP
ETL Process
Aggregates
Reports on recent data and aggregates
BI Server
Reports
Facts Partitions
Meta Data
Copy fact data and dimension
Copy summarized data to Data warehouse
Drill down Data
Connector
Aggregation
Data Interface
Transform fact and dimension data into Big Data schema
Data Store
Big Data Framework (New Components)

Fig. 1.6 The big data business intelligence architecture

1.9 Conclusion

There have been a number of disruptive and decisive transformations in the information technology (IT) discipline in the last five decades. Silently and succinctly IT has moved from a specialized tool to become a greatly and gracefully leveraged tool on nearly every aspect of our lives today. For a long time, IT has been a business enabler and is steadily on the way to becoming the people enabler in the years to unfold. Once upon a time, IT was viewed as a cost center, and now the scene has totally changed to become a profit center for all kinds of organizations across the globe. The role and relevance of IT in this deeply connected and shrunken world are really phenomenal as IT through its sheer strength in seamlessly capturing and embodying all kinds of proven and potential innovations and improvisations is able to sustain its marvelous and mesmerizing journey for the betterment of the human society.

The brewing trends clearly vouch for a digital universe in 2020. The distinct characteristic of the digitized universe is nonetheless the huge data collection (big data) from a variety of sources. This voluminous data production and the clarion call for squeezing out workable knowledge out of the data for adequately empowering the total human society is activating IT experts, engineers, evangelists, and exponents to incorporate more subtle and successful innovations in the IT field. The slogan “more with less” is becoming louder. The inherent expectations from IT for resolving various social, business, and personal problems are on the climb. In this chapter, we have discussed about the digitization paradigm and how next-generation cloud environments are to support and sustain the digital universe.

References

1. Devlin, B.(2012), The Big Data Zoo—Taming the Beasts, a white paper by 9sight Consulting, Retrieved from http://ibmdatamag.com/2012/10/the-big-data-zoo-taming-the-beasts/
2. Kobielus J (2013) The role of stream computing in big data architectures. Retrieved from http://ibmdatamag.com/2013/01/the-role-of-stream-computing-in-big-data-architectures/
3. Persistent (2013) How to enhance traditional bi architecture to leverage big data, a white paper. Retrieved from http://sandhill.com/wp-content/files_mf/persistentsystemswhitepaperenhance_traditionalbi_architecture.pdf

Chapter 2
Demystifying the Traits of Software-Defined Cloud Environments (SDCEs)

Abstract Definitely the cloud journey is on the fast track. The cloud idea got originated and started to thrive from the days of server virtualization. Server machines are being virtualized in order to have multiple virtual machines, which are provisioned dynamically and kept in ready and steady state to deliver sufficient resources (compute, storage, and network) for optimally running any software application. That is, a physical machine can be empowered to run multiple and different applications through the aspect of virtualization. Resultantly, the utilization of expensive compute machines is steadily going up.

This chapter details and describes the nitty-gritty of next-generation cloud centers. The motivations, the key advantages, and the enabling tools and engines along with other relevant details are being neatly illustrated there. An SDCE is an additional abstraction layer that ultimately defines a complete data center. This software layer presents the resources of the data center as pools of virtual and physical resources to host and deliver software applications. A modern SDCE is nimble and supple as per the vagaries of business movements. SECE is, therefore, a collection of virtualized IT resources that can be scaled up or down as required and can be deployed as needed in a number of distinct ways. There are three key components making up SDCEs:

1. Software-defined computing
2. Software-defined networking
3. Software-defined storage

The trait of software enablement of different hardware systems has pervaded into other domains so that we hear and read about software-defined protection, security, etc. There are several useful links in the portal (Sang-Woo et al. “Scalable multi-access flash store for big data analytics” FPGA’14, Monterey, CA, USA, February 26–28, 2014) pointing to a number of resources on the software-defined cloud environments.

2.1 Introduction

Workflow is adopted as an attractive mechanism in a distributed computing environment. A wide range of application domains such as scientific computing, multi-tier Web applications, big data processing, and interpretation applications are

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*, Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_2

represented using workflows. These domain applications consist of multistep computational and data-intensive tasks. Scheduling of these tasks is modeled as workflows. Tasks are linked according to their computational dependencies. These tasks are represented as Directed Acyclic Graphs (DAG) [1]. Many high-performance computing and high-throughput computing workflows are data intensive as well as computation intensive. Many scientific workflows require an HPC environment. Scalability of cloud computing environment supports scientific workflows with the help of virtualization. The dynamic scalable resources are supported by virtual machines in the form of instances in cloud. The process of mapping tasks to computational resources (VMs) for execution is called as "workflow scheduling (WS)."

The workflow scheduling is a task scheduling problem in cloud computing environment, which is proven to be NP-complete. [2]. The on-growing big data applications require HPC resources to execute workflows in a specified amount of time and budget. In cloud computing environment, the end users need to complete their workflows in a specified amount of time with minimum cost. There are many cloud service providers (CSPs) [3], and each provider has different service offerings and supports various parametric objectives (like budget, deadline constraint, energy, security, fault tolerance, etc.) for their service offerings [4–10] as per the end user needs. These parameters play a crucial role in workflow scheduling in a cloud environment. Workflow execution in a cloud environment consists of two main phases:

- Resource provisioning
- Mapping of resources to the tasks

Resource provisioning for the workflows is performed in two ways:

- Static scheduling
- Dynamic scheduling

In a static scheduling environment (clusters and grids), the configurations of the resources are well known in advance. Hence, it is difficult to configure the resources as per user requirements. The dynamic scalability of a cloud environment provides an effective solution to overcome this static scheduling.

In cloud environment, the resources are provisioned dynamically. This is considered as a challenging research problem, due to mapping of appropriate resources among the single CSP, cloud service broker, and private and public cloud. This interesting problem has paved a way for the emergence of various resource scheduling algorithms in recent years.

There are many dynamic scheduling algorithms for mapping of task to the resources. There are many scheduling algorithms in cloud environment, which are classified based on various parameter objectives. The parameter objectives are classified as single-objective, bi-objective, and multi-objective and are considered as per the user needs. The various parameter objectives which are considered in this chapter are:

- Time
- Budget

- Energy
- Resource
- Fault tolerance
- Load balancing
- Security

Thus, virtualization has brought in a number of delectable advancements. Further on, the much-anticipated IT agility and affordability through such virtualization mechanisms are also being realized. It is not only partitioning of physical machines in any data center toward having hundreds of virtual machines in order to fulfil the IT requirements of business activities but also clubbing hundreds of virtual machines programmatically brings forth a large-scale virtual environment for running high-performance computing (HPC) applications. Precisely speaking, the virtualization tenet is leading to the realization of cheaper supercomputing capability. There are other crucial upgrades being brought in by the indomitable virtualization feature, and we will write them too in this book.

Not only servers but also networking components such as firewalls, load balancers, application delivery controllers (ADCs), intrusion detection and prevention systems, routers, switches, gateways, etc. are also getting virtualized in order to deliver these capabilities as network services. The other noteworthy forward-looking movements include the realization of software-defined storage through storage virtualization.

2.2 Reflecting the Cloud Journey

The prime objective of the hugely popular cloud paradigm is to realize highly organized and optimized IT environments for enabling business automation, acceleration, and augmentation. Most of the enterprise IT environments across the globe are bloated, closed, inflexible, static, complex, and expensive. The brewing business and IT challenges are therefore how to make IT elastic, extensible, programmable, dynamic, modular, and cost-effective. Especially with the worldwide businesses cutting down their IT budgets gradually year after year, the enterprise IT team has left with no other options other than to embark on a meticulous and measured journey to accomplish more with less through a host of pioneering and promising technological solutions. Organizations are clearly coming to the conclusion that business operations can run without any hitch and hurdle with less IT resources through effective commoditization, consolidation, centralization, compartmentalization (virtualization and containerization), federation, and rationalization of various IT solutions (server machines, storage appliances, and networking components).

IT operations also go through a variety of technology-induced innovations and disruptions to bring in the desired rationalization and optimization. The IT infrastructure management is also being performed remotely and in an automated manner through the smart leverage of a host of automated IT operation, administration, configuration, software deployment, monitoring, measurement, management, diagnosis (performance, security, etc.), and maintenance tools. The vision of DevOps and NoOps is steadily tending toward the reality. Next-generation data analytics platforms are going to contribute immeasurably in the days to come in decisively automating most of the manual IT operations. Resource failure is being proactively zeroed down through the employment of various analytical capabilities, and the necessary countermeasures are being considered and performed proactively and promptly to ensure business continuity (BC). The application of analytics methods on all kinds of operational, log, performance, security, interaction, and transaction data emanating from every data center resource comes handy in ensuring the optimized service delivery without any hitch and hurdle. The optimal utilization of expensive IT resources is to be guaranteed with the number of advancements in various IT infrastructure management technologies and tools.

With the evolutionary and revolutionary traits of cloud computing, there is a major data center optimization and transformation. The acts of simplification and standardization for achieving IT industrialization are drawing a lot of attention these days. The various IT resources such as memory, disk storage, processing power, and I/O consumption are critically and cognitively monitored, measured, and managed toward their utmost utilization. The pooling and sharing of IT solutions and services are being given the paramount importance toward the strategic IT optimization. Also having a dynamic pool of computing, storage, and network resources enables IT service providers, as well as enterprise IT teams, to meet up any kind of spikes and emergencies in resource needs for their customers and users.

Even with all the unprecedented advancements in the cloud landscape, there are a plenty of futuristic and fulsome opportunities and possibilities for IT professors and professionals to take the cloud idea to its next level in its long journey. Therefore, the concept of software-defined cloud environments (SDCEs) is gaining a lot of accreditation these days. Product vendors, cloud service providers, system integrators, and other principal stakeholders are so keen to have such advanced and acclaimed environments for their clients, customers, and consumers. The right and relevant technologies for the realization and sustenance of software-defined cloud environments are fast maturing and stabilizing, and hence the days of SDCEs are not too far away. This chapter is specially crafted for expressing and exposing all the appropriate details regarding the eliciting and elaborating of the architectural components of SDCEs.

2.2.1 Elucidating the Cloudification Process

The mesmerizing cloud paradigm has become the mainstream concept in IT today and its primary and ancillary technologies are simply flourishing due to the overwhelming acceptance and adoption of the cloud theory. The cloudification movement has blossomed these days, and most of the IT infrastructures and platforms along with business applications are being methodically remedied to be cloud-ready in order to reap all the originally envisaged benefits of the cloud idea. The new buzzword of cloud enablement has caught up fast, and there are collaborative initiatives to unearth techniques, best practices, patterns, metrics, products, and other enablers to understand the cloud fitment and to modernize IT assets and software applications to be cloud oriented.

The virtualization technique has put in a firm and fabulous foundation for the runaway success of cloud computing. Especially server machines are being logically partitioned to carve out a few highly insulated virtual machines (VMs). Then there are a number of standard-compliant and industry-strength automation tools for resource provisioning, configuration, orchestration, monitoring and management, software deployment, and delivery. A 360-degree view of IT infrastructural components through an integrated dashboard is the new normal. Thus, powerful tools play out a very interesting and inspiring role in making cloud pervasive, persuasive, and penetrative. Most of the manual activities associated with the establishment of IT infrastructures, software installation, IT administration and operation, IT services management, and maintenance are being automated through an assortment of technologies. The concept of DevOps is very enticing these days in order to ensure the incredible requirements of IT agility, adaptivity, and affordability.

Automation through templates, patterns, and tools is becoming a common affair in IT lately and substantially reduces human errors. The processes are synchronized to be lean yet efficient. Domain-specific languages (DSLs) are being brought in to bring the required automation. Platforms are being readied to accelerate IT management, governance, and enhancement. There are standards such as OpenStack and their optimal implementations in order to enforce resource portability, interoperability, accessibility, scalability, live-in migration, etc. That is, the distributed deployment of compute instances and storage appliances under the centralized management is the key differentiator for the prodigious success of cloud computing.

Technology Choice is Critical – There are several competent yet contrasting technologies in the IT space today, and hence the selection of implementation technologies has to be strategically planned and carefully played out. Not only the technologies but also the methodologies need to be smartly carried out. In other words, the technology embarkation and usage have to be done with all seriousness and sagacity otherwise, even if the technologies chosen might be sound, yet projects would not see the originally emphasized success. Further on, the history clearly says that many technologies emerged and disappeared from the scene without contributing anything substantial due to the lack of inherent strengths and sagacity.

(continued)

Very few technologies could survive and contribute copiously for a long time. Primarily, the intrinsic complexity toward technologies' all-around utilization and the lack of revered innovations are being touted as the chief reasons for their abject and abysmal failure and the subsequent banishment into thin air. Thus, the factors such as the fitment/suitability, adaptability, sustainability, simplicity, and extensibility of technologies ought to be taken into serious consideration while deciding technologies and tools for enterprise-scale, transformational, and mission-critical projects. The cloud technology is being positioned as the best-in-class technology in the engrossing IT domain with all the necessary wherewithal, power, and potential for handsomely and hurriedly contributing to the business disruption, innovation, and transformation needs. Precisely speaking, the cloud idea is the aggregation of several proven techniques and tools for realizing the most efficient, elegant, and elastic IT infrastructure for the ensuing knowledge era.

2.2.2 The IT Commoditization and Compartmentalization

The arrival of cloud concepts has brought in remarkable changes in the IT landscape that in turn leads in realizing big transitions in the delivery of business applications and services and in the solid enhancement of business flexibility, productivity, and sustainability. Formally cloud infrastructures are centralized, virtualized, automated, and shared IT infrastructures. The utilization rate of cloud infrastructures has gone up significantly. Still, there are dependencies curtailing the full usage of expensive IT resources. Employing the decoupling technique among various modules to decimate all kinds of constricting dependencies, more intensive and insightful process automation through orchestration and policy-based configuration, operation, management, delivery, and maintenance, attaching external knowledge bases, is widely prescribed to achieve still more IT utilization to cut costs remarkably.

Lately, the aroma of commoditization and compartmentalization is picking up. These two are the most important ingredients of cloudification. Let us begin with the commoditization technique.

- *The Commoditization of Compute Machines* – The tried and time-tested abstraction aspect is being recommended for fulfilling the commoditization need. There is a technological maturity as far as physical/bare metal machines getting commoditized through partitioning. The server commoditization has reached a state of semblance and stability. Servers are virtualized, containerized, shared across many clients, publicly discovered, and leveraged over any network, delivered as a service, billed for the appropriate usage, automatically provisioned, composed toward large-scale clusters, monitored, measured, and managed through tools, performance tuned, made policy-aware, automatically scaled up and out based on brewing user, data, and processing needs, etc. In short, cloud servers are being made workloads-aware. However, that is not the case with networking and storage portions.

- *The Commoditization of Networking Solutions* – On the networking front, the propriety and expensive network switches and routers and other networking solutions in any IT data centers and server farms are consciously commoditized through a kind of separation. That is, the control plane gets abstracted out, and hence, the routers and switches have only the data forwarding plane. That means, there is less intelligence into these systems; thereby, the goal of commoditization of network elements is technologically enabled. The controlling intelligence embedded inside various networking solutions is adroitly segregated and is being separately developed and presented as a software controller. This transition makes routers and switches dumb as they lose out their costly intelligence.

  Also, this strategically sound segregation comes handy in interchanging one with another one from a different manufacturer. The vendor lock-in problem simply vanishes with the application of the widely dissected and deliberated abstraction concept. Now with the controlling stake in its pure software form, incorporating any kind of patching in addition to configuration and policy changes in the controlling module can be done quickly in a risk-free and rapid manner. With such a neat and nice abstraction procedure, routers and switches are becoming commoditized entities. There is fresh business and technical advantages as the inflexible networking in present-day IT environments is steadily inching toward to gain the venerable and wholesome benefits of the commoditized networking.
- *The Commoditization of Storage Appliances* – Similar to the commoditization of networking components, all kinds of storage solutions are being commoditized. There are a number of important advantages with such transitions. In the subsequent sections, readers can find more intuitive and informative details on this crucial trait. Currently, commoditization is being realized through the proven abstraction technique.

Thus commoditization plays a very vital role in shaping up the cloud idea. For enhanced utilization of IT resources in an affordable fashion and for realizing software-defined cloud environments, the commoditization techniques are being given more thrusts these days.

The compartmentalization is being realized through the virtualization and containerization technologies. There are several comprehensive books on Docker-enabled containerization in the market, and hence we skip the details of containerization, which is incidentally being touted as the next best thing in the cloud era.

As indicated above, virtualization is one of the prime compartmentalization techniques. As widely accepted and articulated, virtualization has been in the forefront in realizing highly optimized, programmable, managed, and autonomic cloud environments. Virtualization leads to the accumulation of virtualized and software-defined IT resources, which are remotely discoverable, network-accessible, critically assessable, interoperable, composable, elastic, easily manageable, individually maintainable, centrally monitored, and expertly leveraged. The IT capabilities are being given as a service, and hence we often come across the pronouncement "IT as a Service." There is a movement toward the enigma of granting every single IT resource as a service. With the continued availability of path-breaking technologies, resource provisioning is getting automated, and this will result in a new concept of "Resource as a Service (RaaS)."

Bringing in the much-discoursed modularity in order to enable programmable IT infrastructures, extracting and centralizing all the embedded intelligence via robust and resilient software, distributed deployment, centralized management, and federation are being touted as the viable and venerable course of actions for attaining the originally envisaged successes. That is, creating a dynamic pool of virtualized resources, allocating them on demand to accomplish their fullest utilization, charging them for the exact usage, putting unutilized resources back to the pool, monitoring, measuring, and managing resource performance, etc. are the hallmarks of next-generation IT infrastructures. Precisely speaking, IT infrastructures are being software-defined to bring in much-needed accessibility, consumability, malleability, elasticity, and extensibility.

On-demand IT has been the perpetual goal. All kinds of IT resources need to have the inherent capability of preemptively knowing users' as well as applications' IT resource requirements and accordingly fulfill them without any instruction, interpretation, and involvement of human resources. IT resources need to be scaled up and down based on the changing needs so that the cost can be under control. That is, perfect provisioning of resources is the mandate. Overprovisioning raises up the pricing, whereas underprovisioning is a cause for performance degradation worries. The cloud paradigm transparently leverages a number of software solutions and specialized tools in order to provide scalability of applications through resource elasticity. The expected dynamism in resource provisioning and deprovisioning has to become a core and concrete capability of clouds.

That is, providing right-sized IT resources (compute, storage, and networking) for all kinds of business software solutions is the need of the hour. Users increasingly expect their service providers' infrastructures to deliver these resources elastically in response to their changing needs. There is no cloud service infrastructure available today capable of simultaneously delivering scalability, flexibility, and high operational efficiency. The methodical virtualization of every component of a cloud center ultimately leads to software-defined environments.

2.3 Visualizing the Future

1. *Automated Analytics Through Cognitive Computing* – The abovementioned transitions result in massive volumes of poly-structured data. We have been bombarded with a variety of solutions and services in order to realize a number of hitherto unknown analytical capabilities. The analytical platforms facilitate the

extraction of actionable insights out of data heaps. There are big, fast, streaming, and IoT data, and there are batch, real-time, and interactive processing methods. Herein, we need to feed the system with the right and relevant data along with the programming logic on how to process the data and present the insights squeezed out.

However, the future beckons for automated analytics in the sense that we just feed the data and the platform creates viable and venerable patterns, models, and hypotheses in order to discover and disseminate knowledge. The unprecedented growth of cognitive computing is to bring the desired automation so that data gets converted into information and insights casually and cognitively.

2. *Cognitive Clouds* – The cloud journey is simply phenomenal. It started with server virtualization, and we are now moving toward software-defined cloud centers. The originally envisioned goal is to have highly optimized and organized ICT infrastructures and resources for hosting and delivering any kind of software applications. The ICT resource utilization is picking up fast with all the innovations and improvisations in the cloud field. The future belongs to real-time and cognitive analytics of every data emanating out of cloud environments. The log, security, performance, and other operational, transactional, and analytical data can be captured and subjected to a variety of investigations in order to establish dynamic capacity planning, adaptive task/workflow scheduling and load balancing, workload consolidation, and optimization, resource placement, etc. The machine learning (ML) algorithms are bound to come handy in predicting and prescribing the valid steps toward enhanced ICT utilization while fulfilling the functional as well as nonfunctional needs of business applications and IT services.
3. *No Servers, but Services* – The ICT infrastructure operations and management activities such as resource provisioning, load balancing, firewalling, software deployment, etc. are being activated and accelerated through a bevy of automation tools. The vision of NoOps is steadily seeing the reality. There is very less intervention, instruction, interpretation, and involvement of humans in operating and managing IT. People can focus on their core activities blissfully. On the other hand, we are heading toward the serverless architecture. The leading cloud service providers (CSPs) provide the serverless computing capabilities through the various offerings (IBM OpenWhisk, AWS Lambda, Microsoft Azure Functions, and Google Cloud Functions). We are fiddling with bare metal (BM) servers, virtual machines (VMs) through hardware virtualization, and now containers through OS virtualization. The future is for virtual servers. That is, appropriate compute, storage, and network resources get assigned automatically for event-driven applications in order to make way for serverless computing.
4. *Analytics and Applications at the InterCloud of Public, Private, and Edge/Fog Clouds* – Clouds present an illusion of infinite resources, and hence big, fast, streaming, and IoT data analytics are being accomplished in on-premise as well as off-premise, online, and on-demand cloud centers. However, with the faster maturity and stability of device cluster/cloud formation mechanisms due to the faster spread of fog/edge devices in our personal and professional environments,

real-time capture, processing, and action are being achieved these days. That is, the newer capability of edge analytics through edge device clouds for producing real-time insights and services is being widely accepted and accentuated.

5. *Secure Distributed Computing Through Blockchain* – It is a widely expressed concern that security is the main drawback of distributed computing. Similarly, the topmost concern of cloud computing is again the same security aspect. The blockchain technology, which is very popular in financial industries, is now being tweaked and leveraged for other industry segments**.** The blockchain is a sort of public "ledger" of every transaction that has ever taken place. There is no centralized authority, but it is a kind of peer-to-peer (P2P) network of distributed parties to arrive at a consensus, and this consensus is entered into the ledger to be accessed by anyone at a later point in time. It is computationally infeasible for a single actor (or anything less than the majority consensus) to go back and modify history. Moving away from a single decision-maker to multiple decision enablers toward the impenetrable and unbreakable security of any kind of transaction across a myriad of industry verticals is the game-changing breakthrough of the blockchain technology, which is immensely penetrative and participative. There are different viable and venerable use cases across industry segments being considered and empowered by the unlimited power of blockchain technology. There may be salivating convergence among multiple technology domains such as cloud environments, blockchains, artificial intelligence (AI), robotics, self-driving vehicles, cognitive analytics, and insurance domains in the days ahead.

On concluding, the various technological evolutions and revolutions are remarkably enhancing the quality of human lives across the world. Carefully choosing and smartly leveraging the fully matured and stabilized technological solutions and services toward the much-anticipated and acclaimed digital transformation are the need of the hour toward the safe, smarter, and sustainable planet.

2.4 The Emergence of Software-Defined Cloud Environments (SECEs)

We have discussed the commoditization tenet above. Now the buzzword of software-defined everything is all over the place as a fulfilling mechanism for next-generation cloud environments. As widely accepted and accentuated, software is penetrating into every tangible thing in order to bring in decisive and deterministic automation. Decision-enabling, activating, controlling, routing, switching, management, governance, and other associated policies and rules are being coded in software form in order to bring in the desired flexibilities in product installation, administration, configuration, customization, etc. In short, the behavior of any IT products (compute, storage, and networking) is being defined through software. Traditionally all the right and relevant intelligence are embedded into IT systems. Now those insights are being detached from those systems and run in a separate appliance or in virtual machines or in bare metal servers. This detached controlling machine could work with multiple IT systems. It is easy and quick to bring in modifications to the policies in a software controller rather on the firmware, which is embedded inside IT systems. Precisely speaking, deeper automation and software-based configuration, controlling, and operation of hardware resources are the principal enablers behind the longstanding vision of software-defined infrastructure (SDI).

A software-defined infrastructure is supposed to be aware and adaptive to the business needs and sentiments. Such infrastructures are automatically governed and managed according to the business changes. That is, the complex IT infrastructure management is automatically accomplished in consonance with the business direction and destination. Business goals are being literally programmed in and spelled in a software definition. The business policies, compliance and configuration requirements, and other critical requirements are etched in a software form. It is a combination of reusable and rapidly deployable patterns of expertise, recommended configurations, etc. in order to run businesses on the right path. There are orchestration templates and tools, cloud management platforms such as OpenStack, automated software deployment solutions, configuration management and workflow scheduling solutions, etc. in order to accelerate and automate resource provisioning, monitoring, management, and delivery needs. These solutions are able to absorb the abovementioned software definitions and could deliver on them perfectly and precisely.

The SDI automatically orchestrates all its resources to meet the varying workload requirements in near real time. Infrastructures are being stuffed with real-time analytics through additional platforms such as operational, log, performance, and security analytics. As enunciated above, the SDI is agile, highly optimized and organized, and workload-aware. The agility gained out of SDI is bound to propagate and penetrate further to bring the much-needed business agility. The gap between the business expectations and the IT supplies gets closed down with the arrival of software-defined infrastructures. SDI comprises not only the virtualized servers but also virtualized storages and networks.

Software-Defined Cloud Environments vs. Converged Infrastructure (CI) A converged infrastructure is typically a single box internally comprising all the right and relevant hardware (server machines, storage appliance, and network components) and software solutions. This is a being touted as a highly synchronized and optimized IT solution for faster application hosting and delivery. CI is an integrated approach toward data center optimization to substantially minimize the lingering compatibility issues between server, storage, and network components. This gains prominence because it is able to ultimately reduce the costs for cabling, cooling, power, and floor space. CI, a renowned turnkey IT solution, also embeds the software modules for simplifying and streamlining all the management, automation, and orchestration needs. In other words, CI is a kind of appliance specially crafted by a single vendor or by a combination of IT infrastructure vendors. For example, a server vendor establishes a kind of seamless linkage with storage and network product vendors to come out with new-generation CI solutions to speed up the process of IT infrastructure setup, activation, usage, and management.

Hyper-converged infrastructure (HCI) is an extension of the proven CI approach for the highly visible cloud era. The CI vendor abstracts computer, networking, and storage resources from the physical hardware and bundles a virtualization software solution with their CI offerings. Hyper-converged vendors may also provide additional functionality for cloud bursting or disaster recovery and allow administrators to manage both physical and virtual infrastructures (on-premise or off-premise) in a federated manner with a single pane of glass. The CIs and HCIs:

- Accelerate the time to market
- Dramatically reduce downtime
- Simplify IT and respond faster to business demands
- Reduce your total cost of ownership

An SDCE expands this by providing abstraction, pooling, automation, and orchestration across product components from many vendors. This can leverage already running servers and other IT solutions resulting in higher return on investment.

2.5 The Major Building Blocks of Software-Defined Cloud Environments (SDCEs)

Software-defined infrastructures are the key ingredients of SDCEs. That is, an SDCE encompasses software-defined compute, storage, and networking components. The substantially matured server virtualization leads to the realization of software-defined compute machines. Highly intelligent hypervisors (alternatively recognized as virtual machine monitors (VMMs)) act as the perfect software solution to take care of the creation, provisioning, deprovisioning, live-in migration, decommissioning of computing machines (virtual machines and bare metal servers), etc. Most of the servers across leading cloud centers are virtualized, and it is clear that the server virtualization is reaching a state of stability. In a sense, the SDCE is simply the logical extension of server virtualization. The server virtualization dramatically maximizes the deployment of computing power. Similarly, the SDCE does the same for all of the resources needed to host an application, including storage, networking, and security.

In the past, provisioning a server machine to host an application took weeks of time. Today a VM can be provisioned in a few minutes. Even containers can be provisioned in a few seconds. That is the power of virtualization and containerization. This sort of speed and scale being made possible through virtualization platforms is being extended to other IT resources. That is, the whole cloud center is getting fully virtualized in order to tend toward the days of software-defined clouds.

In SDCEs, all IT resources are virtualized so they can be automatically configured and provisioned and made ready to install applications without any human intervention, involvement, and interpretation. Applications can be operational in minutes; thereby, the time to value has come down sharply. The IT cost gets reduced significantly. There are a number of noteworthy advancements in the field of server virtualization in the form of a host of automated tools, design and deployment patterns, easy-to-use templates, etc. The cloud paradigm became a famous and fantastic approach for data center transformation and optimization because of the unprecedented success of server virtualization. This riveting success has since then penetrated into other important ingredients of data centers.

IT resources that are virtualized thereby are extremely elastic, remotely programmable, easily consumable, predictable, measurable, and manageable. With the comprehensive yet compact virtualization sweeping each and every component of data centers, the goals of distributed deployment of various resources but centrally monitored, measured, and managed are nearing the reality. Server virtualization has greatly improved data center operations, providing significant gains in performance, efficiency, and cost-effectiveness by enabling IT departments to consolidate and pool computing resources. Considering the strategic impacts of 100% virtualization, we would like to focus on network and storage virtualization methods in the sections to follow.

2.5.1 Network Virtualization

Server virtualization has played a pivotal and paramount role in cloud computing. Through server virtualization, the goals of on-demand and faster provisioning besides the flexible management of computing resources are readily and rewardingly fulfilled. Strictly speaking, server virtualization also includes the virtualization of network interfaces from the operating system (OS) point of view. However, it does not involve any virtualization of the networking solutions such as switches and routers. The crux of the network virtualization is to derive multiple isolated virtual networks from sharing the same physical network. This paradigm shift blesses virtual networks with truly differentiated capabilities to coexist on the same infrastructure and to bring forth several benefits toward data center automation and transformation.

Further on, VMs across geographically distributed cloud centers can be connected to work together to achieve bigger and better things for businesses. These virtual networks can be crafted and deployed on demand and dynamically allocated for meeting differently expressed networking demands of different business applications. The functionalities of virtual networks are decisively varying. That is, virtual networks not only come handy in fulfilling the basic connectivity requirement but also are capable of getting tweaked to get a heightened performance for specific workloads. Figure 2.1 vividly illustrates the difference between server and network virtualization.

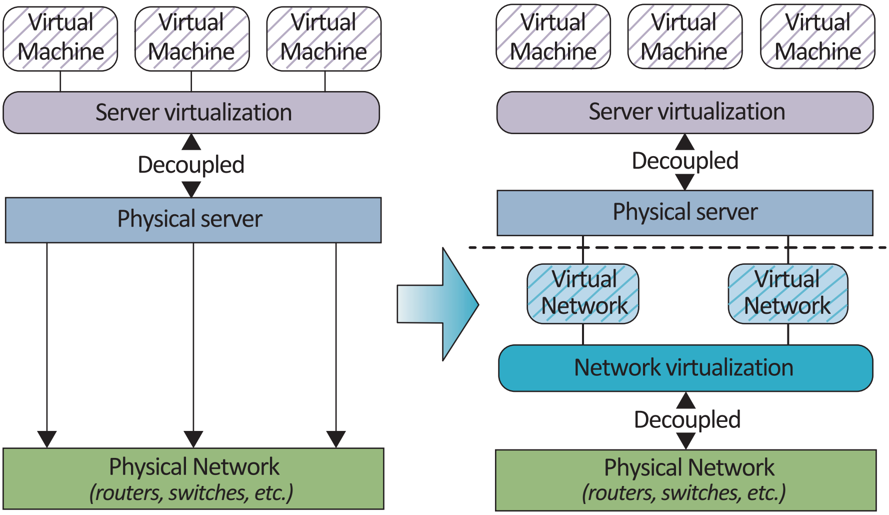


Fig. 2.1 Depicting the differences between server and network virtualization

2.5.2 *Network Functions Virtualization (NFV)*

There are several network functions such as load balancing, firewalling, routing, switching, etc. in any IT environment. The idea is to bring forth the established virtualization capabilities into the networking arena so that we can have virtualized load balancing, firewalling, etc. The fast-emerging domain of network functions virtualization aims to transform the way that network operators and communication service providers architect and operate communication networks and their network services.

The today's IT environment is exceedingly dynamic with the steady incorporation of cloud technologies. New virtual machines can be spun up in minutes and can migrate between physical hosts. Application containers are also emerging fast in order to speed up application composition, packaging, and shipping across data centers. The network remains relatively static and painstakingly slow in the sense that it is an error-prone provisioning process to provide network connectivity for applications. Data center networks have to facilitate the movement of applications between computing servers within data centers as well as across data centers. There is also a need for layer 2 VLAN extensions. In today's traditional LAN/WAN design, the extension of VLANs and their propagation within data centers is not an easy affair. Ensuring all redundant links, in addition to switches, are properly configured can be a time-consuming operation. This can introduce errors and risks. With the trends such as big data, bring your own devices (BYOD) and data- and process-intensive videos, the IT infrastructures, especially networks, are under immense pressure.

Network virtualization provides a supple and nimble network environment and is being touted as the best-in-class solution approach for tackling all the abovementioned trends in the IT landscape. Network functions virtualization (NFV) is getting a lot of attention these days, and network service providers have teamed up well to convince their product vendors to move away from special-purpose equipment and appliances toward software-only solutions. These software solutions run on commodity servers, storages, and network elements such as switches, routers, application delivery controllers (ADCs), etc. By embracing the NFV technology, communication and cloud service providers could bring down their capital as well as operational costs significantly. When the power consumption goes down, the heat dissipation is also bound to go down. Also the cost of employing expert resources for administering and operating special equipment is bound to come down significantly, as well as time to market for conceiving and concretizing newer and premium services. Due to its software-driven approach, NFV also allows service providers to achieve a much higher degree of operational automation and to simplify operational processes such as capacity planning, job scheduling, workload consolidation, VM placement, etc.

In an NFV environment, the prominent operational processes, such as service deployment, on-demand allocation of network resources such as bandwidth, failure detection, on-time recovery, and software upgrades, can be easily programmed and executed in an automated fashion. This software-induced automation brings down the process time to minutes rather than weeks and months. There is no need for the operational team to personally and physically visit remote locations to install, configure, diagnose, and repair network solutions. Instead, all kinds of network components can be remotely monitored, measured, and managed.

In short, it is all about consolidating diverse network equipment types (firewall, switching, routing, ADC, EPC, etc.) onto industry-standard x86 servers using virtualization. The immediate and strategic benefits include the operational agility, which could empower business agility, autonomy, and affordability.

Network virtualization helps worldwide enterprises achieve major advancements in simplicity, speed, and security by clinically automating and simplifying many of the data center operations and processes. NV also contributes immensely to reducing the network provisioning time from weeks to minutes. It helps to achieve higher operational productivity by automating a variety of manual processes. NV comes handy in placing and moving workloads across data centers. Finally, the network security gets a boost.

2.5.3 Software-Defined Networking (SDN)

Software-defined networking is quite a recent concept that is to disaggregate the traditional vertically integrated networking stack in order to improve network flexibility and manageability. SDN represents a bevy of technologies that facilitate the data, control, and management planes of the network to be loosely coupled to be distantly accessible through APIs, independently extensible and evolving. These APIs also facilitate the development of a rich new set of network applications and services from a wider range of sources, including independent developers, value-added resellers (VARs), and user organizations themselves.

The brewing technology trends indicate that networks and network management are bound to change once for all. Today's data centers (DCs) extensively use physical switches and appliances that haven't yet been virtualized and are statically and slowly provisioned. Further on, there is a current environment mandate for significant and certified expertise in operating each vendor's equipment. The networking solutions also lack an API ecosystem toward facilitating remote discovery and activation. In short, the current situation clearly points out the absence of programmable networks. It is quite difficult to bring in the expected automation (resource provisioning, scaling, etc.) on the currently running, inflexible, monolithic, and closed network and connectivity solutions. The result is the underutilization of expensive network equipment. Also, the cost of employing highly educated and experienced network administrators is definitely on the higher side. Thus, besides bringing in a bevy of pragmatic yet frugal innovations in the networking arena, the expressed mandate for substantially reducing the capital as well as the operational expenses being incurred by the traditional network architecture is clearly playing in the minds of technical professionals and business executives.

As the virtualization principle has been contributing immensely to server consolidation and optimization, the idea of network virtualization has picked up in the recent past. The virtualization aspect on the networking side takes a different route compared to the matured server virtualization. The extraction and centralization of network intelligence embedded inside all kinds of network appliances such as routers, switches, etc. into a centralized controller esthetically bring in a number of strategic advantages for data centers. The policy-setting, configuration, and maneuvering activities are being activated through software libraries that are modular, service-oriented, and centralized in a controller module, and hence the new terminology "software-defined networking" (SDN) has blossomed and is hugely popular. That is, instead of managing network assets separately using separate interfaces, they are controlled collectively through a comprehensive, easy-to-use and fine-grained interface. The application programming interface (API) approach has the intrinsic capability of putting a stimulating and sustainable foundation for all kinds of IT resources and assets to be easily discoverable, accessible, usable, and composable. Simplistically speaking, the aspect of hardware infrastructure programming is seeing the reality, and thereby the remote manipulations and machinations of various IT resources are gaining momentum.

The control plane manages switch and routing tables, while the forwarding plane actually performs the layer 2 and 3 filtering, forwarding, and routing. In short, SDN (Fig. 2.2) decouples the system that makes decisions about where traffic is sent (the control plane) from the underlying system that forwards traffic to the selected destination (the data plane). This well-intended segregation leads to a variety of innova-

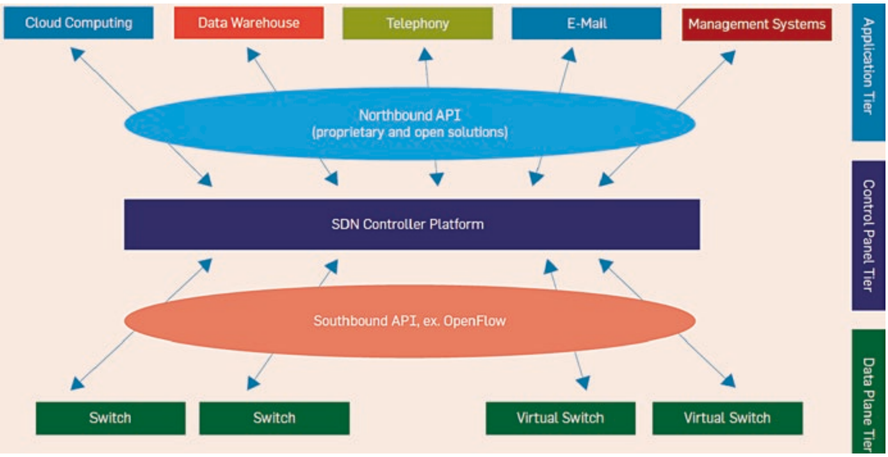


Fig. 2.2 The macro-level SDN architecture

tions and inventions. Therefore, standards-compliant SDN controllers provide a widely adopted API ecosystem, which can be used to centrally control multiple devices in different layers. Such an abstracted and centralized approach offers many strategically significant improvements over traditional networking approaches. For instance, it becomes possible to completely decouple the network's control plane and its data plane as illustrated in Fig. 2.3. The control plane runs in a cluster setup and can configure all kinds of data plane switches and routers to support business expectations as demanded. That means data flow is regulated at the network level in an efficient manner. Data can be sent where it is needed or blocked if it is deemed a security threat.

A detached and deft software implementation of the configuration and controlling aspects of network elements also means that the existing policies can be refurbished, whereas newer policies can be created and inserted on demand to enable all the associated network devices to behave in a situation-aware manner. As we all know, policy establishment and enforcement are the proven mechanisms to bring in the required versatility and vitality in network operations. If a particular application's flow unexpectedly needs more bandwidth, SDN controller proactively recognizes the brewing requirement in real time and accordingly reroute the data flow in the correct network path. Precisely speaking, the physical constraints are getting decimated through the software-defined networking. If a security appliance needs to be inserted between two tiers, it is easily accomplished without altering anything at the infrastructure level. Another interesting factor is the most recent phenomenon of "bring your own device (BYOD)." All kinds of employees' own devices can be automatically configured, accordingly authorized, and made ready to access the enterprise's network anywhere any time.

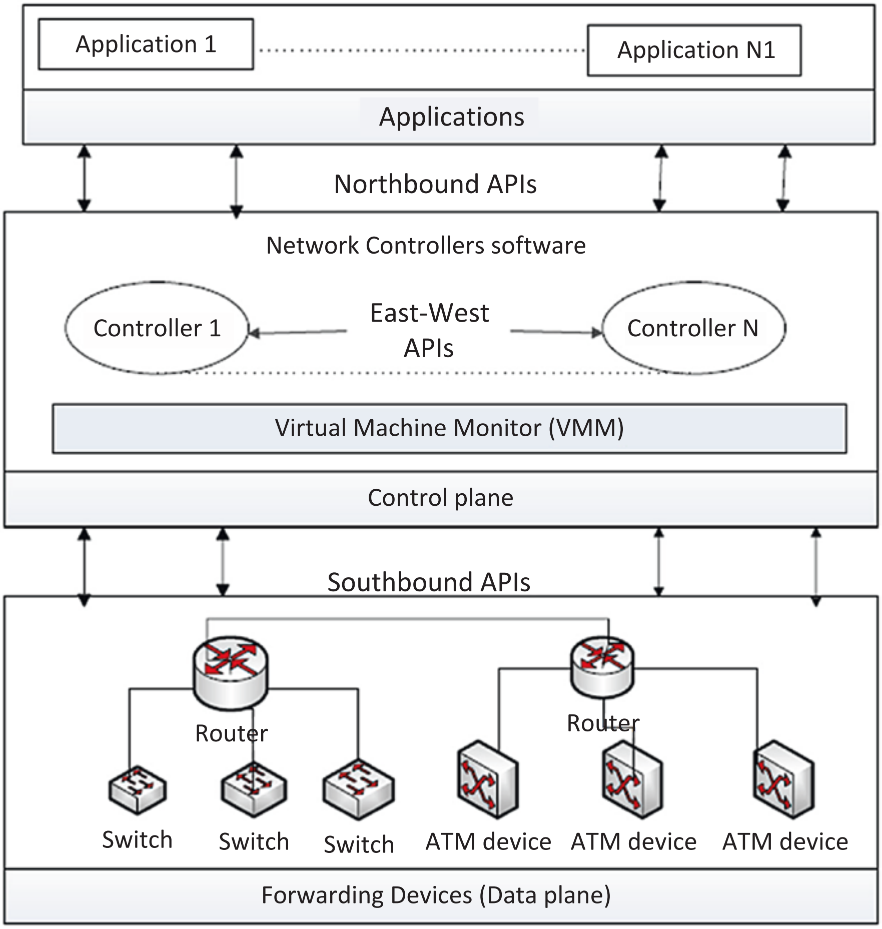


Fig. 2.3 The separation of control and data planes

2.5.4 The Key Motivations for SDN

In the IT world, there are several trends mandating the immediate recognition and sagacious adoption of SDN. Software-defined cloud environments (SDCEs) are being established in different cool locations across the globe to provide scores of orchestrated cloud services to worldwide businesses and individuals over the Internet on a subscription basis. Application and database servers besides integration middleware solutions are increasingly distributed, whereas the governance and the management of distributed resources are being accomplished in a centralized manner to avail the much-needed single point of view (SPoV). Due to the hugeness of data centers, the data traffic therefore internally as well as externally is exploding these days. Flexible traffic management and ensuring "bandwidth on demand" are the principal requirements.

The consumerization of IT is another gripping trend. Enterprise users and executives are being increasingly assisted by a bevy of gadgets and gizmos such as smartphones, laptops, tablets, wearables, etc. in their daily chores. As enunciated elsewhere, the "bring your own device (BYOD)" movement requires enterprise networks to inherently support policy-based adjustment, amenability, and amelioration to support users' devices dynamically. Big data analytics (BDA) has a telling effect on IT networks, especially on data storage and transmission. The proprietary nature of network solutions from worldwide product vendors also plays a sickening role in traditional networks, and hence there is a clarion call for bringing in necessary advancements in the network architecture. Programmable networks are therefore the viable and venerable answer to bring in the desired flexibility and optimization in highly complicated and cumbersome corporate networks. The structural limitations of conventional networks are being overcome with network programming. The growing complexity of traditional networks leads to stasis. That is, adding or releasing devices and incorporating network-related policies are really turning out to be a tough affair at the current setup.

As per the leading market watchers, researchers, and analysts, SDN marks the largest business opportunity in the networking industry since its inception. Recent reports estimate the business impact tied to SDN could be as high as $35 billion by 2018, which represents nearly 40% of the overall networking industry. The future of networking will rely more and more on software, which will accelerate the pace of innovation incredibly for networks as it has in the computing and storage domains (explained below). SDN has all within to transform today's static and sick networks into calculative, competent, and cognitive platforms with the intrinsic intelligence to anticipate and allocate resources dynamically. SDN brings up the scale to support enormous data centers and the virtualization needed to support workload-optimized, converged, orchestrated, and highly automated cloud environments. With its many identified advantages and astonishing industry momentum, SDN is on the way to becoming the new norm and normal not only for cloud but also corporate networks. With the next-generation hybrid and federated clouds, the role of SDN for fulfilling network function virtualization (NFV) is bound to shoot up.

In short, SDN is an emerging architecture that is agile, adaptive, cheaper, and ideal for network-intensive and dynamic applications. This architecture decouples the network control and forwarding functions (routing), enabling the network control to become directly programmable and the underlying infrastructure to be abstracted for applications and network services, which can treat the network as a logical or virtual entity.

2.5.5 The Need of SDN for the Cloud

Due to a number of enterprise-wide benefits, the adoption rates of cloud paradigm have been growing. However, the networking aspect of cloud environments has typically not kept pace with the rest of the architecture. There came a number of enhancements such as network virtualization (NV), network function virtualization (NFV), and software-defined networking (SDN). SDN is definitely the comprehensive and futuristic paradigm. With the explosion of computing machines (both virtual machines as well as bare metal servers) in any cloud centers, the need for SDN is sharply felt across. Networks today are statically provisioned, with devices that are managed at a box-level scale and are underutilized. SDN enables end-to-end-based network equipment provisioning, reducing the network provisioning time from days to minutes, and distributing flows more evenly across the fabric allowing for better utilization.

2.6 The Distinct Benefits of Software-Defined Networking

The benefits of SDN are definitely diversified and gradually enlarging. SDN is being seen as a blessing for cloud service providers (CSPs), enterprise data centers, telecommunication service providers, etc. The primary SDN benefits are the following:

The Centralized Network Control and Programmability As we discussed above, the gist of the SDN paradigm is to separate the control function from the forwarding function. This separation resultantly facilitates the centralized management of the network switches without requiring physical access to the switches. The IT infrastructure deployments and management are therefore gaining the most benefits as SDN controller is capable of creating, migrating, and tearing down VMs without requiring manual network configurations. This feature maximizes the value of large-scale server virtualization.

Dynamic Network Segmentation VLANs provide an effective solution to logically group servers and virtual machines at the enterprise or branch network level. However, the 12-bit VLAN ID cannot accommodate more than 4096 virtual networks, and this presents a problem for mega data centers such as public and private clouds. Reconfiguring VLANs is also a daunting task as multiple switches and routers have to be reconfigured whenever VMs are relocated. The SDN's support for centralized network management and network element programmability allows highly flexible VM grouping and VM migration.

High Visibility of VMs The virtual hypervisor switch and all the VMs running in a physical server use only one or two NICs to communicate with the physical network. These VMs are managed by server management tools and hence not visible to network management tools. This lacuna makes it difficult for network designers and administers to understand the VM movement. However, SDN-enabled hypervisor switches and VMs alleviate this visibility problem.

Capacity Utilization With centralized control and programmability, SDN easily facilitates VM migration across servers in the same rack or across clusters of servers in the same data center or even with servers in geographically distributed data centers. This ultimately leads to automated and dynamic capacity planning that in turn significantly increments physical server utilization.

Network Capacity Optimization The classic tri-level design of data center networks consisting of core, aggregation, and access layer switches (North-South design) is facing scalability limits and poses inefficiencies for server-to-server (East-West) traffic. There are innovative solutions such as link aggregation, multi-chassis link aggregation, top-of-rack (ToC) switches, and layer 2 multipath protocols. These are able to fulfill load-balancing, resiliency, and performance requirements of dense data centers. However, these are found to be complex and difficult to manage and maintain. The SDN paradigm enables the design and maintenance of network fabrics that span across multiple data centers.

Distributed Application Load Balancing With SDN, it is possible to have the load-balancing feature that chooses not only compute machines but also the network path. It is possible to have geographically distributed load-balancing capability.

OpenFlow is a kind of SDN protocol that specifies an API for programming the flow tables of a network switch. Traditionally, these flow tables could not be programmed remotely or by third parties. Network switches typically included a proprietary network operating system (NOS) and native programs which controlled the flow tables. With OpenFlow, the switch only manages flow tables. The OS and control programs get hosted and executed on a commodity server. This arrangement removes constraints on control software. That is, the control software can be programmed using any language and run on any operating system. Also, the control software can run on the virtual machine (VM) and bare metal (BM) server. With containerization sweeping the cloud space, we can expect the control software to be containerized.

With the embedded intelligence getting abstracted and extracted out of switches, the network switches are getting dumber, cheaper, and more capable. Precisely speaking, the OpenFlow enables the intelligence required to manage LANs and WANs to run in software while pushing the physical execution down to the switches. It is all about software-defined control and management of physical hardware. Further on, additional capabilities can be realized from hardware systems. That is, networks can perform faster, route data based on business needs, and enable Internet-scale application communication through SDN.

On summarizing, SDN is the definite game changer for next-generation IT environments. SDN considerably eliminates network complexity in the midst of multiple and heterogeneous network elements. All kinds of network solutions are centrally configured and controlled to eliminate all kinds of dependencies-induced constrictions and to realize their full potential. Network capabilities are provisioned on demand at the optimal level to suit application requirements. In synchronization with other infrastructural models appropriately, the on-demand, instant-on, autonomic, and smart computing goals are easily delivered.

2.7 Accentuating Software-Defined Storage (SDS)

This is the big data era. Huge volumes of data are being generated, captured, cleansed, and crunched in order to extract actionable insights due to the realization that big data leads to big insights. Data is the new fuel for the world economy. IT has to be prepared accordingly and pressed into service in order to garner, transmit, and stock multi-structured and massive data. Software-defined storage gives enterprises, organizations, and governments a viable and venerable mechanism to effectively address this explosive data growth.

We are slowly yet steadily getting into the virtual world with the faster realization of the goals allied with the concept of virtual IT. The ensuing world is leaning toward the vision of anytime-anywhere access to information and services. This projected transformation needs a lot of perceivable and paradigm shifts. Traditional data centers were designed to support specific workloads and users. This has resulted in siloed and heterogeneous storage solutions that are difficult to manage, provision newer resources to serve dynamic needs, and finally to scale out. The existing setup acts as a barrier for business innovations and value. Untangling this goes a long way in facilitating instant access to information and services.

Undoubtedly storage has been a prominent infrastructural module in data centers. There are different storage types and solutions in the market. In the recent past, the unprecedented growth of data generation, collection, processing, and storage clearly indicates the importance of producing and provisioning of better and bigger storage systems and services. Storage management is another important topic not to be sidestepped. We often read about big, fast, and even extreme data. Due to an array of technology-inspired processes and systems, the data size, scope, structure, and speed are on the climb. For example, digitization is an overwhelming worldwide trend and trick gripping every facet of human life; thereby, the digital data is everywhere and continues to grow at a stunning pace. Statisticians say that every day, approximately 15 petabytes of new data is being generated worldwide, and the total amount of digital data doubles approximately every 2 years. The indisputable fact is that machine-generated data is larger compared to man-generated data. The expectation is that correspondingly there have to be copious innovations in order to cost-effectively accommodate and manage big data.

As we all know, storage appliances are embedded with sophisticated software stacks. And there is storage management software to perform management tasks such as moving files and provisioning volumes. It is not just the abstraction of the embedded intelligence and the creation of a software module outside the storage appliances and arrays. The essence of SDS is to leverage the distributed computing techniques in designing and setting up storage appliances. That is, using commodity hardware and innovative storage optimization techniques for space and performance efficiencies, accessibility, manageability, etc. is being touted as the next-generation software-defined storage. This kind of advanced storage approach is inherently capable of tackling explosive data growth, bringing down storage costs through commodity hardware, leveraging already invested storage arrays, providing geographical data replication, etc. Further on, the SDS is famous for simplified and scalable storage management experience. A variety of data access protocols can be accommodated in the SDS environment. There are capital and operational cost savings with SDS, and the experience of nimbler, supple, scalable, and streamlined storage solutions is also accrued and availed.

Due to the distributed nature, the scale-out is inherently elastic in the sense that additional commodity nodes for storage can be obtained quickly if there is a fluctuation in data storage requirements. When the demand comes down, all the additionally provisioned storage systems can be brought into the storage pool in order to be ready to serve. This is a prime reason for the runaway success of the SDS concept. The traditional storages primarily support the scale-up method, which is not in tune with the increasingly dynamic nature of IT environments.

Traditionally, storage administrators pre-allocate logical unit number (LUN) addresses of storage in shared storage hardware. This is for making any idle capacity to be available for virtual machine disks when virtual machines are created. Several different LUNs can be created and kept ready to accommodate varying performance and business requirements. With software-defined storage, virtual workloads are being decoupled from physical storage. Software-defined storage will pool all storage capacity into a data plane and assign storage by using a policy-based control plane that is informed with the performance characteristics of the underlying storage targets. The result is application-centric or virtual machine-centric control of pooled storage resources.

Software-defined storage (SDS) is a relatively new concept, and its popularity is surging due to the abundant success attained in software-defined compute and networking areas. As explained above, SDS is a part and parcel of the vision behind the establishment and sustenance of software-defined cloud environments (SDCEs). With the virtualization concept penetrating and piercing through every tangible resource, the storage industry also gets inundated by that powerful trend. Software-defined storage is a kind of enterprise-class storage that uses a variety of commoditized and, therefore, cheap hardware with all the important storage and management functions being extricated and performed using an intelligent software controller. With such a clean separation, SDS delivers automated, policy-driven, and application-aware storage services through an orchestration of the underlining storage infrastructure. That is, we get a dynamic pool of virtual storage resources to be picked up dynamically and orchestrate them accordingly to be presented as an appropriate storage solution. Unutilized storage resources could be then incorporated into the pool for serving other requests. All kinds of constricting dependencies on storage solutions simply vanish with such storage virtualization. All storage modules are commoditized, and hence the cost of storage is to go down with higher utilization. In a nutshell, storage virtualization enables storage scalability, replaceability, substitutability, and manageability.

An SDS solution remarkably increases the flexibility by enabling organizations to use nonproprietary standard hardware and, in many cases, leverage existing storage infrastructures as a part of their enterprise storage solution. Additionally, organizations can achieve a massive scale with an SDS by adding heterogeneous hardware components as needed to increase capacity and improve performance in the solution. Automated, policy-driven management of SDS solutions helps drive cost and operational efficiencies. As an example, SDS manages important storage functions including information lifecycle management (ILM), disk caching, snapshots, replication, striping, and clustering. In a nutshell, these SDS capabilities enable you to put the right data in the right place, at the right time, with the right performance, and at the right cost automatically.

Unlike traditional storage systems such as SAN and NAS, SDS simplifies scale-out with relatively inexpensive standard hardware while continuing to manage storage as a single enterprise-class storage system. SDS typically refers to software that manages the capture, placement, protection, and retrieval of data. SDS is characterized by a separation of the storage hardware from the software that manages it. SDS is a key enabler modernizing traditional, monolithic, inflexible, costly, and closed data centers toward software-defined data centers that are highly extensible, open, and cost-effective. The promise of SDS is that separating the software from the hardware enables enterprises to make storage hardware purchase, deployment, and operation independent from concerns about over- or underutilization or interoperability of storage resources.

Cloud-Based Big Data Storage Object storage is the recent phenomenon. Object-based storage systems use containers/buckets to store data known as objects in a flat address space instead of the hierarchical, directory-based file systems that are common in the block and file-based storage systems. Nonstructured and semi-structured data are encoded as objects and stored in containers. Typical data includes emails, pdf files, still and dynamic images, etc. Containers store the associated metadata (date of creation, size, camera type, etc.) and the unique Object ID. The Object ID is stored in a database or application and is used to reference objects in one or more containers. The data in an object-based storage system is typically accessed by

HTTP using a web browser or directly through an API like REST (representational state transfer). The flat address space in an object-based storage system enables simplicity and massive scalability. But the data in these systems can't be modified, and every refresh gets stored as a new object. Object-based storage is predominantly used by cloud services providers (CSPs) to archive and backup their customers' data.

Analysts estimate that more than 2 million terabytes (or 2 exabytes) of data are created every day. The range of applications that IT has to support today spans everything from social computing, big data analytics, mobile, enterprise and embedded applications, etc. All the data for all those applications has got to be made available to mobile and wearable devices, and hence data storage acquires an indispensable status. As per the main findings of Cisco's global IP traffic forecast, in 2016, global IP traffic will reach 1.1 zettabytes per year or 91.3 exabytes (1 billion gigabytes) per month, and by 2018, global IP traffic will reach 1.6 zettabytes per year or 131.9 exabytes per month. IDC has predicted that cloud storage capacity will exceed 7 exabytes in 2014, driven by a strong demand for agile and capex-friendly deployment models. Furthermore, IDC had estimated that by 2015, big data workloads will be one of the fastest-growing contributors to storage in the cloud. In conjunction with these trends, meeting service-level agreements (SLAs) for the agreed performance is a top IT concern. As a result, enterprises will increasingly turn to flash-based SDS solutions to accelerate the performance significantly to meet up emerging storage needs.

2.8 The Key Characteristics of Software-Defined Storage (SDS)

SDS is characterized by several key architectural elements and capabilities that differentiate it from the traditional infrastructure.

Commodity Hardware With the extraction and centralization of all the intelligence embedded in storage and its associated systems in a specially crafted software layer, all kinds of storage solutions are bound to become cheap, dumb, off-the-shelf, and hence commoditized hardware elements. Not only the physical storage appliances but also all the interconnecting and intermediate fabric is to become commoditized. Such segregation goes a long way in centrally automating, activating, and adapting the full storage landscape.

Scale-Out Architecture Any SDS setup ought to have the capability of ensuring a fluid, flexible, and elastic configuration of storage resources through software. SDS facilitates the realization of storage as a dynamic pool of heterogeneous resources; thereby, the much-needed scale-out requirement can be easily met. The traditional architecture hinders the dynamic addition and release of storage resources due to the extreme dependency. For the software-defined cloud environments, storage scalability is essential to have a dynamic, highly optimized, and virtual environment.

Resource Pooling The available storage resources are pooled into a unified logical entity that can be managed centrally. The control plane provides the fine-grained visibility and the control to all available resources in the system.

Abstraction Physical storage resources are increasingly virtualized and presented to the control plane, which can then configure and deliver them as tiered storage services.

Automation The storage layer brings in extensive automation that enables it to deliver one-click and policy-based provisioning of storage resources. Administrators and users request storage resources in terms of application need (capacity, performance, and reliability) rather than storage configurations such as RAID levels or physical location of drives. The system automatically configures and delivers storage as needed on the fly. It also monitors and reconfigures storage as required to continue to meet SLAs.

Programmability In addition to the inbuilt automation, the storage system offers fine-grained visibility and control of underlying resources via rich APIs that allows administrators and third-party applications to integrate the control plane across storage, network, and compute layers to deliver workflow automation. The real power of SDS lies in the ability to integrate it with other layers of the infrastructure to build end-to-end application-focused automation.

The maturity of SDS is to quicken the process of setting up and sustaining software-defined environments for the tactic as well as the strategic benefits of cloud service providers as well as the consumers at large.

2.8.1 Software-Defined Wide Area Networking (SD-WAN)

Application performance is very crucial for worldwide enterprises. There is a spurt in the number of applications being hosted in enterprise IT environments, and applications are being relied upon for performing mission-critical functions. Networking plays a very vital role in shaping up the application performance. The public Internet is the mainstream communication technology and tool. MPLS is another popular but expensive networking solution for delivering highly reliable and faster connectivity for business automation.

The public Internet simply does not have the inherent wherewithal to support the global delivery of modern applications due to the unpredictable latency and congestion-based packet loss. The other prominent connectivity methods such as MPLS are not optimized for cloud service access. They are expensive and consume longer time to get deployed. As the cloud environments across the globe are being stuffed with SaaS applications to be delivered worldwide, the need for better alternatives surfaces, and hence there is a surging popularity for SD-WAN that provides enterprise-grade connectivity and guarantees consistently fast performance for on-premise and SaaS-based applications, regardless of where they are located.

There are network providers and products such as Aryaka's Software-Defined Network Platform to ensure optimized, software-defined network connectivity and application acceleration to globally distributed enterprises. The key capabilities include the following:

SD-WAN Is a Kind of Global Private L2 Network This network is capable of bypassing the congested public Internet through a global private network (using an SDN/NFV framework) that delivers consistent latencies and negligible packet loss to provide predictable application performance to users.

TCP Optimization This software-defined network delivers end-to-end latency mitigation, packet loss mitigation, and enhanced congestion management that supercharges TCP applications for faster delivery.

Compression The SD-WAN solutions typically reduce the amount of data transferred across links using compression algorithms for higher throughput and faster application performance.

Deduplication This software solution eliminates the transmission of redundant data sent more than once over the network. This reduces bandwidth consumption up to 98% when transmitting data between locations and improves application performance.

SaaS Acceleration This specialized solution integrates cloud-hosted SaaS applications in a secure fashion. The attached application acceleration proxies in the SD-WAN solution could deliver up to 40x faster performance globally for cloud/SaaS applications.

Cloud Connectors There are multiple cloud service providers (CSPs), and the SD-WAN solution is stuffed with a number of connectors to facilitate private and dedicated connectivity to the leading public clouds such as IBM Bluemix, AWS, Azure, and Google Clouds.

There are other features and functionalities being incorporated in the new-generation SD-WAN solutions in order to make the cloud idea more popular and pervasive.

2.9 The Key Benefits of Software-Defined Cloud Environments (SDCEs)

The new technologies have brought in highly discernible changes in how data centers are being operated to deliver both cloud-enabled and cloud-native applications as network services to worldwide subscribers. Here are a few important implications (business and technical) of SDCEs.

The consolidation and centralization of commoditized, easy-to-use and easy-to-maintain, and off-the-shelf server, storage, and network hardware solutions obviate the need for having highly specialized and expensive server, storage, and networking components in IT environments. This cloud-inspired transition brings down the capital as well as operational costs sharply. The most important aspect is the introduction and incorporation of a variety of policy-aware automated tools in order to quickly provision, deploy, deliver, and manage IT systems. There are other mechanisms such as templates, patterns, and domain-specific languages for automated IT setup and sustenance. Hardware components and application workloads are being provided with well-intended APIs in order to enable remote monitoring, measurement, and management of each of them. The APIs facilitate the system interoperability. The direct fallout here is that we can arrive at highly agile, adaptive, and affordable IT environments. The utilization of hardware resources and applications goes up significantly through sharing and automation. Multiple tenants and users can avail the IT facility comfortably for a cheaper price. The cloud technologies and their smart leverage ultimately ensure the system elasticity, availability, and security along with application scalability.

Faster Time to Value The notion of IT as a cost center is slowly disappearing, and businesses across the globe have understood the strategic contributions of IT in ensuring the mandated business transformation. IT is being positioned as the most competitive differentiator for worldwide enterprises to be smartly steered in the right direction. However, there is an insistence for more with less as the IT budget is being consistently pruned every year. Thus enterprises started to embrace all kinds of proven and potential innovations and inventions in the IT space. That is, establishing data centers locally or acquiring the right and relevant IT capabilities from multiple cloud service providers (CSPs) is heavily simplified and accelerated. Further on, resource provisioning, application deployment, and service delivery are automated to a greater extent, and hence it is easier and faster to realize the business value. In short, the IT agility being accrued through the cloud idea translates into business agility.

Affordable IT By expertly pooling and assigning resources, the SDCEs greatly maximize the utilization of the physical infrastructures. With enhanced utilization through automation and sharing, the cloud center brings down the IT costs remarkably while enhancing the business productivity. The operational costs come down due to tools-supported IT automation, augmentation, and acceleration.

Eliminating Vendor Lock-In Today's data center features an amazing array of custom hardware for storage and networking requirements such as routers, switches, firewall appliances, VPN concentrators, application delivery controllers (ADCs), storage controllers, intrusion detection, and prevention components. With the storage and network virtualization, the above functions are performed by software running on commodity x86 servers. Instead of being locked into the vendor's hardware, IT managers can buy commodity servers in quantity and use them for running the network and storage controlling software. With this transition, the perpetual vendor lock-in issue gets simply solved and surmounted. The modifying source code is quite easy and fast, policies can be established and enforced, software-based activation and acceleration of IT network and storage solutions are found to be simple, supple and smart, etc.

Less Human Intervention and Interpretation SDCEs are commoditized and compartmentalized through abstraction, virtualization, and containerization mechanisms. As accentuated above, there are infrastructure management platforms, integration, orchestration engines, integrated brokerage services, configuration, deployment and delivery systems, service integration and management solutions, etc. in order to bring in deeper and decisive automation. That is, hitherto manually performed tasks are getting automated through toolsets. This enablement sharply lessens the workloads of the system, storage, and service administrators. All kinds of routine, redundant, and repetitive tasks are getting automated on a priority basis. The IT experts, therefore, can focus on their technical expertise to come up with a series of innovations and inventions that subsequently facilitate heightened business resiliency and robustness.

Hosting a Range of Applications All kinds of operational, transactional, and analytical workloads can be run on SDCEs, which is emerging as the comprehensive yet compact and cognitive IT infrastructure to ensure business operations at top speed, scale, and sagacity. Business continuity, backup and archival, data and disaster recovery, high availability, and fault tolerance are the other critical requirements that can be easily fulfilled by SDCEs. As we expectantly move into the era of big data, real-time analytics, mobility, cognitive computing, social networking, web-scale systems, the Internet of Things (IoT), artificial intelligence, deep learning, etc., the SDCEs are bound to play a very stellar and sparkling role in the days ahead.

Distributed Deployment and Centralized Management IT resources and business applications are being extremely distributed these days by giving considerations for cost, location, performance, risk, etc. However, a 360-degree view through a single pane of glass is required in order to have a firm and fine grip on each of the assets and applications. The centralized monitoring, measurement, and management are the most sought-after feature for any SDCE. The highly synchronized and unified management of various data center resources is getting fulfilled through SDCE capabilities.

Streamlined Resource Provisioning and Software Deployment There are orchestration tools for systematic and swift provisioning of servers, storages, and network components. As each resource is blessed with RESTful or other APIs, the resource provisioning and management become simpler. Policies are the other important ingredient in SDCEs in order to have intelligent operations. As we all know, there are several configuration management tools, and in the recent past, with the culture of DevOps spreads widens overwhelmingly, there are automated software deployment solutions. Primarily, orchestration platforms are for infrastructure, middleware, and database installation, whereas software deployment tools take care of application installation.

Containerized Platforms and Workloads With the unprecedented acceptance of Docker-enabled containerization and with the growing Docker ecosystem, there is a wave of containerization across the data centers and their operations. Packaged, home-grown, customized, and off-the-shelf business applications are being containerized; IT platforms, database systems, and middleware are getting containerized through the open-source Docker platform; and IT infrastructures are increasingly presented as a dynamic pool of containers. Thus SDCEs are the most appropriate one for containerized workloads and infrastructures.

Adaptive Networks As inscribed above, an SDCE comprises network virtualization that in turn guarantees network function virtualization (NFC) and software-defined networking (SDN). Network bandwidth resource can be provisioned and provided on demand as per the application requirement. Managing networking solutions such as switches and routers remains a challenging assignment for data center operators. In an SDC, all network hardware in the data center is responsive to a centralized controlling authority, which automates network provisioning based on defined policies and rules. A dynamic pool of network resources comes handy in fulfilling any varying network requirements.

Software-Defined Security Cloud security has been a challenge for cloud center professionals. Hosting mission-critical applications and storing customer, confidential, and corporate information on cloud environments are still a risky affair. Software-defined security is emerging as the viable and venerable proposition for ensuring unbreakable and impenetrable security for IT assets, business workloads, and data sources. Policy-based management, the crux of software-defined security, is able to ensure the much-required compliance with security policies and principles. SDCE is innately stuffed with software-defined security capabilities.

Green Computing SDCEs enhance resource utilization through workload consolidation and optimization, VM placement, workflow scheduling, dynamic capacity planning, and management. Energy-awareness is being insisted as the most vital parameter for SDCEs. When the electricity consumption goes down, the heat dissipation too goes down remarkably; thereby, the goal of green and lean computing gets fulfilled. This results in environment sustainability through reduced release of harmful greenhouse gasses.

In summary, applications that once ran on static, monolithic, and dedicated servers are today hosted in software-defined, policy-aware, consolidated, virtualized, automated, and shared IT environments that can be scaled and shaped to meet brewing demands dynamically. Resource allocation requests that took days and weeks to fulfill now can be accomplished in hours or even in minutes. Virtualization and containerization have empowered data center operations, enabling enterprises to deploy commoditized and compartmentalized servers, storages, and network solutions that can be readily pooled and allocated to a fast-shifting application demand.

On concluding, an SDCE delivers two prominent outcomes. Firstly, it enables businesses to shift their time, treasure, and talent toward innovation and growth by encapsulating agility, adaptivity, and efficiency into IT offerings and operations.

2.10 Conclusion

The aspect of IT optimization is continuously getting rapt and apt attention from technology leaders and luminaries across the globe. A number of generic, as well as specific, improvisations are being brought in to make IT aware and adaptive. The cloud paradigm is being touted as the game changer in empowering and elevating IT to the desired heights. There have been notable achievements in making IT being the core and cost-effective enabler of both personal as well as professional activities. There are definite improvements in business automation, acceleration, and augmentation. Still, there are opportunities and possibilities waiting for IT to move up further.

The pioneering virtualization technology is being taken to every kind of infrastructures such as networking and storage to complete the IT ecosystem. The abstraction and decoupling techniques are lavishly utilized here in order to bring in the necessary malleability, extensibility, and serviceability. That is, all the configuration and operational functionalities hitherto embedded inside hardware components are now neatly identified, extracted and centralized, and implemented as a separate software controller. That is, the embedded intelligence is being developed now as a self-contained entity so that hardware components could be commoditized. Thus, the software-defined computing, networking, and storage disciplines have become the hot topic for discussion and dissertation. The journey of data centers (DCs) to software-defined cloud environments (SDCEs) is being pursued with vigor and rigor. In this chapter, we have primarily focused on the industry mechanism for capturing and collecting requirement details from clients.

References

1. Hadoop Cluster Applications, a white paper by Arista, 2013
2. Chandhini C, Megana LP (2013) Grid Computing-A next level challenge with big data. Int J Sci Eng Res, March 2013
3. Brinker DL, Bain WL (2013) Accelerating Hadoop MapReduce using an in-memory Data Grid, a white paper from ScaleOut Software, Inc
4. White C (2014) Why Big Data in the Cloud?, a white paper by BI Research, January 2014
5. Performance and Scale in Cloud Computing, a white paper by Joyent, 2011
6. Mengjun Xie, Kyoung-Don Kang, Can Basaran (2013) Moim: a Multi-GPU MapReduce Framework
7. Stuart JA, Owens JD (2012) Multi-GPU MapReduce on GPU Clusters
8. The Elephant on the Mainframe, a white paper by IBM and Veristorm, April 2014
9. Olofson CW, Vesset D (2013) The Mainframe as a Key Platform for Big Data and Analytics, a white paper by IDC 2013
10. Sang-Woo Jun, Ming Liu, Kermin Elliott Fleming, Arvind (2014) Scalable multi-access flash store for big data analytics. FPGA'14, February 26–28, 2014, Monterey, CA, USA

Chapter 3
Workflow Management Systems

Abstract Today's scientific application requires tremendous amount of computation-driven as well as data-driven supported resources. The scientific applications are represented as workflows. The workflow management systems are designed and developed to depict the workflows of complex nature. The workflow management systems are able to reliably and efficiently coordinate among various resources in a distributed environment. This chapter describes various workflow management software like Kepler, Taverna, Triana, Pegasus, and Askalon. The architecture and functionalities of these workflow management systems are explained in the following sections.

3.1 Introduction

Scientific workflows are used to model the complex applications with DAG (Directed Acyclic Graph) format with nodes and edges which are easy to express the entire data process with its dependencies. Huge amount of data are consumed and produced during the scientific experiments, which made the workflows data intensive. As complexity of scientific applications increases, the need for using Scientific Workflow Management System also increases to automate and orchestrate the end-to-end processing. In order to process the large-scale data, they need to be executed in a distributed environment such as grid or cloud.

Scientific Workflow Management System is an effective framework to execute and manage massive datasets in computing environment. Several workflow management systems, such as Kepler [1], Taverna [2], Triana [3], Pegasus [4], and Askalon [5], are available, and they are widely used by the researchers in various fields such as astronomy, biology, limnology, geography, computational engineering, and others. The suitable environment for workflow computation, storage provisions, and execution is provided by grid, cluster, and cloud.

Cloud computing is a widely used computing environment comprising of several data centers with its own resources and data, which provides a scalable and on-demand service over the Internet as pay-as-you-go pricing model. This chapter provides a detailed description of widely used scientific workflow management systems.

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_3

3.2 Workflow Management System

Workflow management systems provide tools to define, map, and execute workflow applications. There are numerous WMSs available to support workflow execution in different domains. This section describes some of the significant WMS developed by the research community. Workflow systems develop their own workflow representations to describe workflows. Generally, workflow models are roughly categorized into two types: script-like systems and graphical-based systems. Script-like systems describe workflow using textual programming languages such as Java, Python, Perl, or Ruby. They declare task and their dependencies with a textual specification. Examples of script-based model are GridAnt [6] and Karajan [7]. Graphical-based systems describe workflow with basic graphical elements with nodes and edge. It is easy as compared with the script-based model. The node represents the actual computational tasks and the communication between the tasks are represented with the help of links. Workflows are often created with the dragging and dropping graph elements.

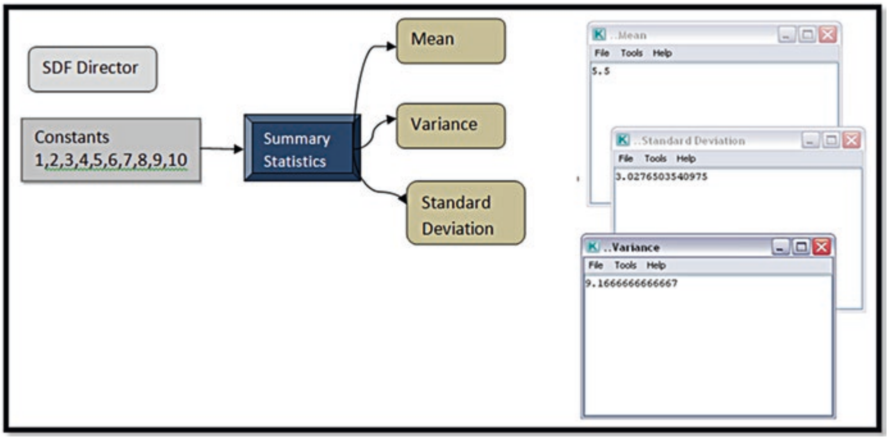
3.3 Kepler

A scientific workflow is an automated process. It combines data and processes in a well-defined set of steps for implementing computational solutions for a given scientific problem [10]. Kepler [9] is a cross project collaboration to develop a scientific workflow system for multiple disciplines that provide a workflow environment in which scientists can design and execute workflows.

Kepler is an open-source Java framework which is developed and maintained by the cross project Kepler collaboration. It is derived from Ptolemy [1]. Kepler is designed to support scientists and researchers to analyze and modeling of a wide range of scientific and engineering applications. Kepler software application is used for analysis and modeling of scientific data. Kepler supports visual representation of the process. With the help of visual representation, it simplifies the effort in creating executable models.

It enables researchers to create executable models by using visual representation. Scientific workflows are created by dragging and dropping of components to the workflow creation area, and the connection is made between the components to construct specific dataflow. Scientific workflows exhibit the flow of data between discrete analysis and modeling components, which is shown in Fig. 3.1. It allows scientist to design and execute scientific workflows either from command line or from graphical user interface.

Kepler uses distributed computing technologies to share workflow and their data around the world with other scientists. It also provides the scientists to access the computing resources, data repositories, and workflow libraries.



```
5.5
```

```
3.0276503540975
```

```
9.1666666666667
```

Fig. 3.1 Scientific workflow representation in Kepler [1]

Kepler is based on directors, which dictate the flow of execution within a workflow. The basic components of Kepler are director, actor, parameter, ports, and relations which are shown in Fig. 3.2.

Director – It is used to represent the different types of components with the model of its computation in a workflow, and it controls the overall execution flow.
Actor – Executes the director instructions, and the composite actor performs the complex operations.
Ports – Actors are connected to each other with the help of ports. An actor may contain one or more ports to produce or consume data to communicate with other actors. Link is the dataflow from one actor port to another.
Relations – Allows user to branch a dataflow.
Parameter – Configurable values attached to a workflow, director, or actors.

Features of Kepler

- Run-time engine and graphical user interface are supported which is helpful for executing the workflows with GUI or CLI.
- Reusability is supported in Kepler; modules and components can be created, saved, and reused for other workflow applications also.
- Native support for parallel processing applications.
- Reusable library contains 350 ready-to-use process components. These components can be easily customized and used for other workflow applications. Kepler supports integration with other applications such as:

  Statistical analyses into Kepler workflows are possible, by integrating it with R and MATLAB.

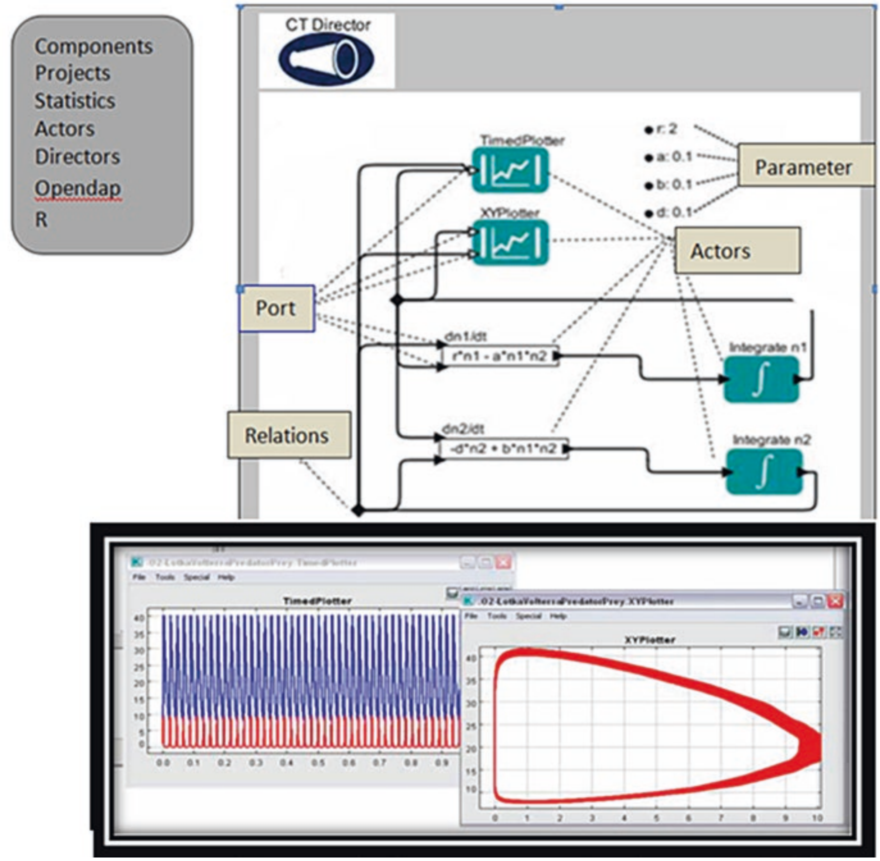


Fig. 3.2 Basic components of Kepler [1]

WSDL defined Web services support for workflows accessing and execution

- Workflows can be uploaded, downloaded, and searched on the Kepler's Component Repository which provides a centralized server.
- The rapid development and scalable distributed execution of bioinformatics workflows in Kepler are the latest version in bioKepler 1.2 consists of new features for bioinformatics applications such as:

  Workflow for BLAST+
  Machine learning
  Updating to Spark 1.5.0 to Spark 1.5.0

3.4 Taverna

Taverna is a Java-based open-source workflow management system created by myGrid team, which used to design and execute scientific workflows. The significant goal of Taverna is to extend support for life science domain such as chemistry,

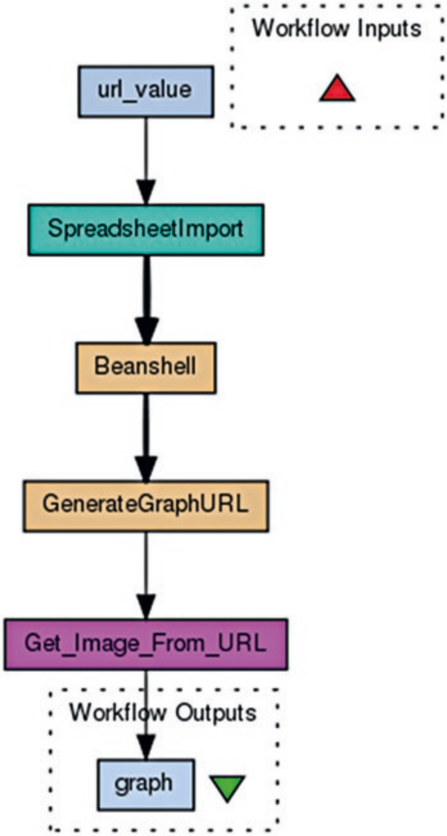


Fig. 3.3 Scientific workflow representation in Taverna (Example using the Spreadsheet Import service to import data from an Excel spreadsheet)

biology, and medicine to execute scientific workflows and support in silicon experimentation, where the experiments are carried through computer simulations with models, which are closely reflecting the real world. It supports Web services, local Java services and API, R scripts, and CSV data files.

Workbench in Taverna supports desktop client application which is a GUI, provides create, edit and run workflows. Taverna Server is used for remote execution. Taverna Player is a web interface for execution of workflows through the Taverna server. Taverna also supports Command Line Tool for executing workflows using Command line instruction. Taverna provides service discovery facility and supports for browsing selected service catalogues. Figure 3.3 represents workflow designed in Taverna. It can access local processes, Web services, and also high-performance computing infrastructures through Taverna PBS plugin.

Taverna utilizes a simple conceptual unified flow language (Scufl) to represent the workflow [11]. Scufl is an XML-based language, which consists of three main entities: processors, data links, and coordination constraints. Processors represent a computational activity in a scientific workflow. Data links and coordination constraints separately represent the data dependencies and control dependencies between two activities.

Graphical SWfMSs combine the efficiency of scientific workflow design and the ease of scientific workflow representation. Desktop-based graphical SWfMSs are typically installed either in a local computer or in a remote server that is accessible through network connection. The local computer or remote server can be connected to large computing and storage resources for large-scale scientific workflow execution.

The data storage module generally exploits database management systems and file systems to manage all the data during workflow execution. Some SWfMSs such as Taverna put intermediate data and output data in a database.

3.5 Triana

Triana is a Java-based scientific workflow system developed at Cardiff University, which uses a visual interface with data analysis tools. Triana has various built-in tools for image manipulation, signal analysis, and others, which also allow researchers to integrate their own tools. Triana has the workflow engine called Triana Controlling Service (TCS) for executing workflows. Figure 3.4 depicts the scientific workflow designed in Taverna.

Triana GUI connects to the TCS either locally or remotely; it runs workflow application and visualizes it locally. When the applications run in batch modes, it logs periodically to check the status of the application. Triana GUI contains the collection of Triana components which allows researchers to create workflow

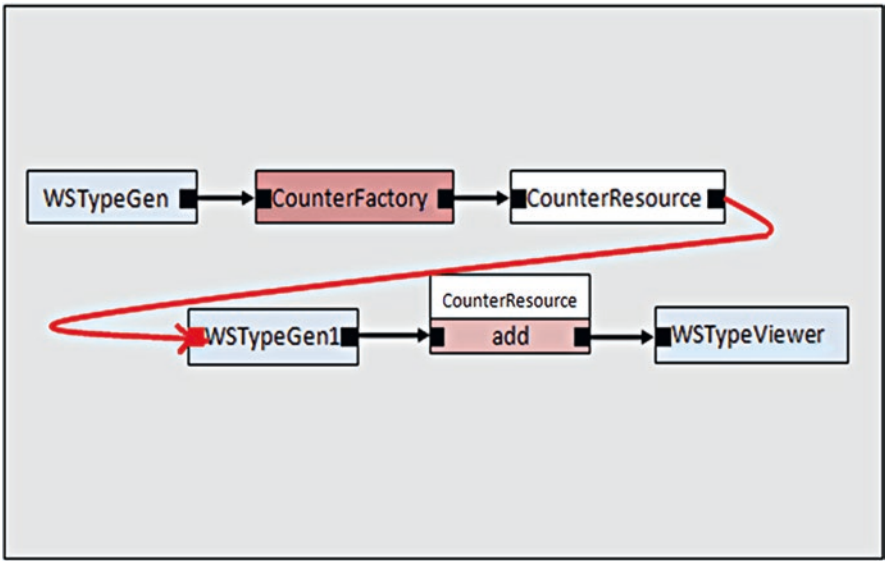


Fig. 3.4 Scientific workflow representation in Triana

applications of the desired behavior. Each component contains the information about input and output data items. Triana also provides the generalized reading and writing interfaces to integrate third-party workflow representation and services within the GUI.

3.6 Pegasus

Pegasus [8] is an open-source workflow management system developed at the University of Southern California, which integrated a number of technologies to execute workflow application in heterogeneous environments such as grid, clusters, and cloud. Pegasus has been used in a number of scientific domains such as bioinformatics, gravitational wave physics, ocean science, and others. The architecture of the Pegasus Workflow Management System is shown in Fig. 3.5.

Pegasus links the scientific domain and the execution environment by mapping the abstract workflow description to an executable workflow description for computation and executes the workflow application in their order of dependencies. Scientific workflow representation in Pegasus is depicted in Fig. 3.6.

Pegasus consists of five major subsystems for performing various actions which include mapper, local execution engine, job scheduler, remote execution engine, and monitoring component.

- Mapper – Generates an executable workflow from the abstract workflows which were provided by the user. Mapper finds an appropriate resource and other

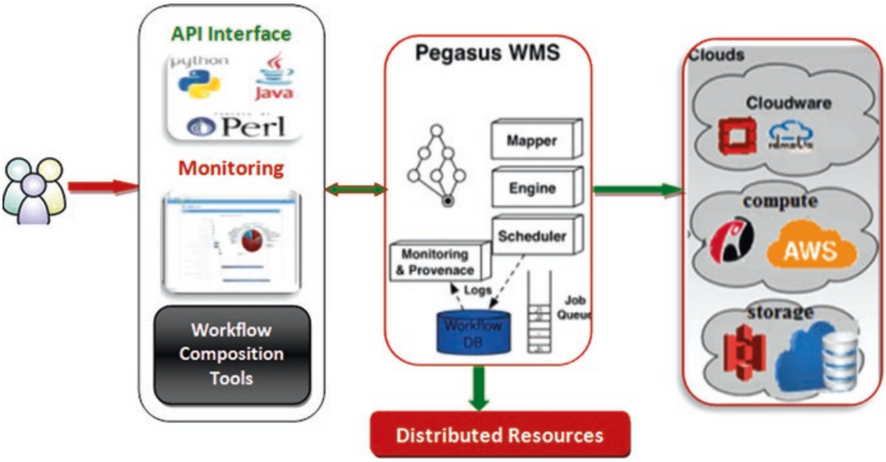


Fig. 3.5 Architecture of Pegasus

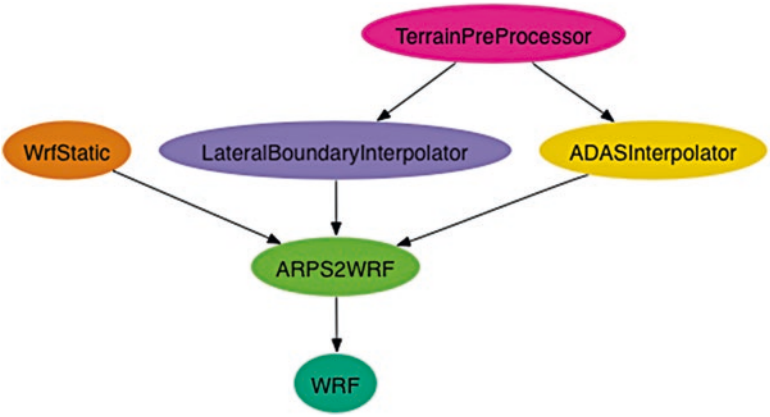


Fig. 3.6 Scientific workflow representation in Pegasus

related components which are required for the workflow execution. It also restructures workflows for optimized performance.

- Local execution engine – Submit the jobs and manage it by tracking the job state and determines when to run a job. Then it submits the jobs to the local scheduling queue.
- Job scheduler – It manages the jobs on both local and remote resources
- Remote execution engine – It manages the execution of jobs on the remote computing nodes.
- Monitoring component – Monitors the workflow and track all the information's and logs. Workflow database collects those information and provides the provenance and performance information. Additionally it notifies the user about the workflow status.

3.7 ASKALON

ASKALON Project is developed at the University of Innsbruck; it provides an ideal environment based on new services, tools, and methodologies for executing workflow applications in cloud and grid environment. Its aim is to simplify the development and optimization of workflow applications. Workflows are described using Abstract Grid Workflow Language (AGWL). It allows researchers to design applications using modeling instead of programming. It provides the optimization of workflow applications based on cost constraints. Many real-world applications such as Wien2k, Invmod, MetwoAG, and GRASIL have been ported in ASKALON. Figure 3.7 represents the scientific workflow representation in ASKALON. The main components of ASKALON are:

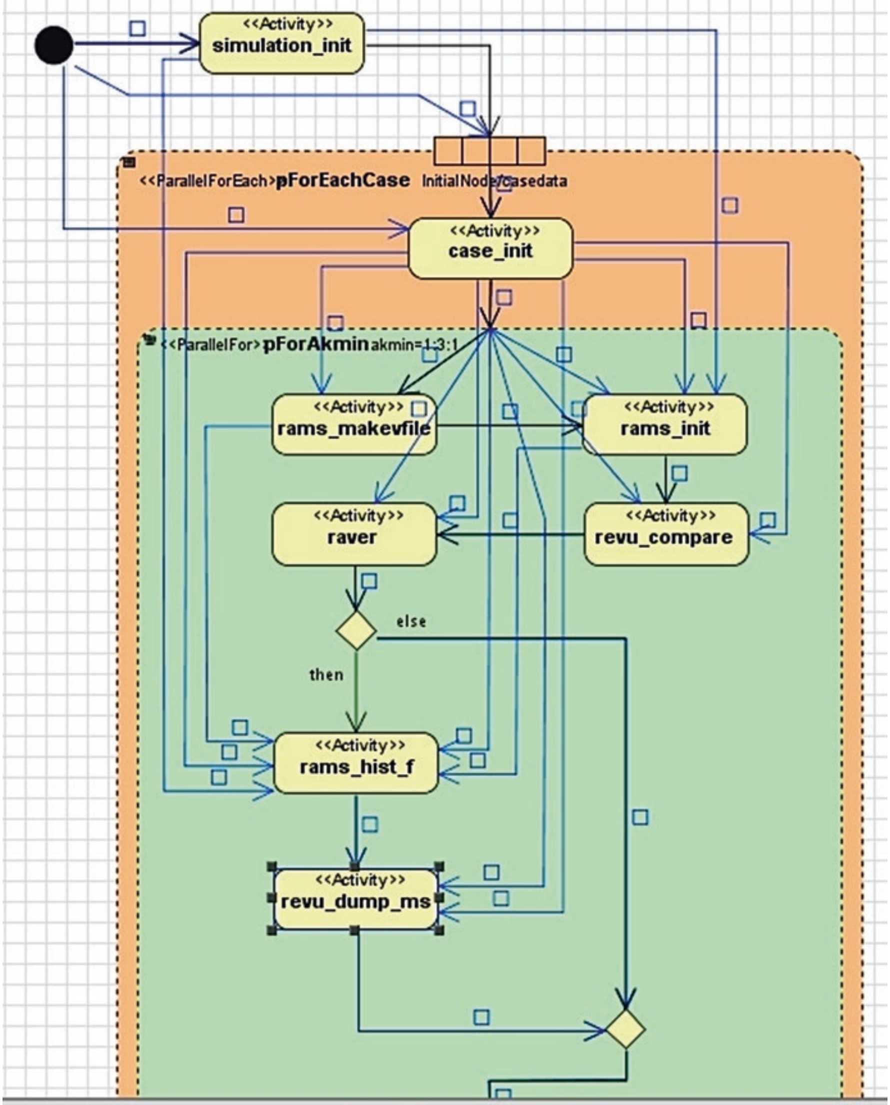


Fig. 3.7 Scientific Workflow representation in ASKALON

- *Resource broker*: Reserves the resources required for the execution of workflow applications.
- *Resource monitoring*: Support for monitoring grid and cloud resources by integration and scaling of resources using new techniques.
- *Information service:* Service for discovering and organizing resources and data.
- *Workflow executer*: Executing workflow applications in remote cloud and grid sites.

- *Scheduler*: Maps workflow applications on to the grid or cloud resources
- *Performance prediction*: Investigates new techniques for the estimation of execution and transfer time and also the resource availability.

3.8 Conclusion

Scientific workflows represent the complex applications as DAG wherein the nodes and edges which are easy to express the entire data process with its dependencies. Workflow management systems provide tools to define, map, and execute workflow applications. This chapter highlights the salient features of most popular workflow management systems such as Kepler, Pegasus, Triana, Taverna, and Askalon. However, the workflows that are created using WMS require following important ingredients for their efficient execution: scheduling algorithm and heterogeneous and scalable computing environment. Chapter 4 explains all of these perspectives by reviewing several notable research works that are available in literature.

References

1. https://kepler-project.org/ (getting started guide version2.5 October 2015)
2. http://www.taverna.org.uk/
3. http://www.trianacode.org/index.php
4. https://pegasus.isi.edu/
5. http://www.askalon.org/
6. von Laszewski G, Hategan M (2005) Java CoG Kit Karajan/Gridant workflow guide. Technical report. Argonne National Laboratory, Argonne, Ill, USA
7. von Laszewski G, Hategan M, Kodeboyina D (2007) Java CoG kit workflow. In: Workflows for e-Science. Springer, New York, pp 143–166
8. Deelman E, Vahi K, Rynge M, Juve G, Mayani R, Ferreira da Silva R, (2016) Pegasus in the Cloud: science automation through workflow technologies, IEEE Internet Comput 20(1):70–76 (Funding Acknowledgements: NSF ACI SI2-SSI 1148515, NSF ACI 1245926, NSF FutureGrid 0910812, NHGRI 1U01 HG006531-01)
9. Ludäscher B, Altintas I, Berkley C, Higgins D, Jaeger E, Jones M, Lee EA, Tao J, Zhao Y (2006) Scientific workflow management and the Kepler System. Concurr Comput Pract Exp 18:1039–1065
10. Altintas I, Barney O, Jaeger-Frank E (2006) Provenance collection support in the Kepler scientific workflow system. International Provenance and Annotation Workshop (IPAW'06)
11. Oinn TM, Addis M, Ferris J, Marvin D, Senger M, Greenwood RM, Carver T, Glover K, Pocock MR, Wipat A, Li P (2004) Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20(17):3045–3054

Chapter 4
Workflow Scheduling Algorithms and Approaches

Abstract Cloud infrastructures typically offer access to boundless virtual resources dynamically provisioned on demand for hosting, running, and managing a variety of mission-critical applications like scientific workflows, big data processing application, business intelligence-based applications, high-performance computing (HTC), and high transaction computing (HTC). Due to the surging popularity of the irresistible cloud idea, there are cloud datacenters spreading across the globe comprising heterogeneous cloud platforms and infrastructures catering to fast-evolving demands of worldwide businesses. The pervasive connectivity has enabled for the unprecedented success of the cloud concept. However, intensive automation is the key to the originally intended success of the cloud paradigm. Researchers across the world are focusing on unearthing powerful and pioneering tools and techniques for automated infrastructure life-cycle management. Similarly there are pathbreaking work-around approaches, algorithms, and architectures for workload consolidation. In short, there are many cloud-related aspects yearning for technologically sound automation, acceleration, and augmentation capabilities.

Efficient scheduling algorithms become mandatory for automated operations of distributed and disparate cloud resources and workloads. The resource scheduling is a dynamic problem, and it is associated with on-demand resource provisioning, fault tolerance support, and hybrid resource scheduling with appropriate Quality of Service, considering time, cost, and budget. This chapter provides the details about various automated solutions for workflow scheduling and also comprehensive survey of various existing workflow scheduling algorithms in the cloud computing environment.

4.1 Introduction

Workflow is adopted as an attractive mechanism in a distributed computing environment. Wide range of application domains such as scientific computing, multi-tier Web applications, high-performance computing applications, big data processing, and interpretation applications are represented using workflows. These domain applications consist of multistep computational and data-intensive tasks. Scheduling of these tasks is modeled as workflows. Tasks are linked according to their computational dependencies. These tasks are represented as Directed Acyclic Graphs

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_4

(DAG) [1]. Many high-performance computing and high-throughput computing workflows are data intensive as well as computation intensive. Many scientific workflows requires HPC environment. Scalability of cloud computing environment supports scientific workflows with the help of virtualization. The dynamic scalable resources are supported by virtual machines in the form of instances in cloud. The process of mapping tasks to computational resources (VMs) for execution is called as "workflow scheduling (WS)."

The workflow scheduling is task scheduling problem in cloud computing environment, which is proven to be NP-complete [2]. The on-growing big data applications require HPC resources to execute workflows in a specified amount of time and budget. In cloud computing environment, the end users need to complete their workflows in a specified amount of time with minimum cost. There are many cloud service providers (CSP) [3], and each provider has different service offerings. CSP supports various parametric objectives (like budget, deadline constraint, energy, security, fault tolerance, etc.) for their service offerings [4–11] as per the end user needs. These parameters play a crucial role in workflow scheduling in cloud environment. Workflow execution in a cloud environment consists of two main phases:

- Resource provisioning
- Resource mapping to the tasks

Resource provisioning for the workflows are performed in two ways:

- Static scheduling
- Dynamic scheduling

In static scheduling environment (clusters and grids), the configurations of the resources are well known in advance. Hence, it is difficult to configure the resources as per user requirements. Dynamic scalability of cloud environment provides an effective solution to overcome this static scheduling.

In cloud environment the resources are provisioned dynamically. This is considered as a challenging research problem, due to mapping of appropriate resources among the single CSP, cloud service broker, and private and public cloud. This interesting problem has paved a way for the emergence of various resource scheduling algorithms in recent years.

There are many dynamic scheduling algorithms for mapping of task to the resources. There are many scheduling algorithms in cloud environment, which are classified based on various parameter objectives. The parameter objectives are classified as single-objective, bi-objective, and multi-objective which are considered as per the user needs. The various parameter objectives which are considered in this chapter are:

- Time
- Budget
- Energy
- Resource
- Fault tolerance

- Load balancing
- Security

The workflow model and its structure are dealt in section 4.2 along with taxonomy of workflow scheduling.

4.2 Workflow Model

Workflow applications are represented as a Directed Acyclic Graph (DAG), in G (T, E) where G is a graph, T represents set of tasks, and E denotes the set of directed edges which represents the dependency between tasks. In an application each task represents the individual task which is measured in million instructions (MI). The task with no precedence task is called as entry task, and the task with no descendant task is called exit task. Workflow size is determined by the total number of tasks. Workflows are used widely in business and scientific applications. Figure 4.1 represents the sample workflow with ten tasks. LIGO, CyberShake, Montage, and GT-FAR are the examples of workflow applications.

Workflow Structure

A structure of a workflow describes the dependency between the tasks in a workflow. The structure can be categorized as sequence, parallel, and choice in a DAG-based structure. In sequence structure workflow will be executed in a serial manner; parallel structure executes the workflow tasks concurrently. In choice structure the workflows are executed in serial as well as parallel. In Fig. 4.1, the tasks T1, T2, T4, T7, T9, and T10 represent the sequence workflow structure, whereas T5 and T6 represent the parallel workflow structure.

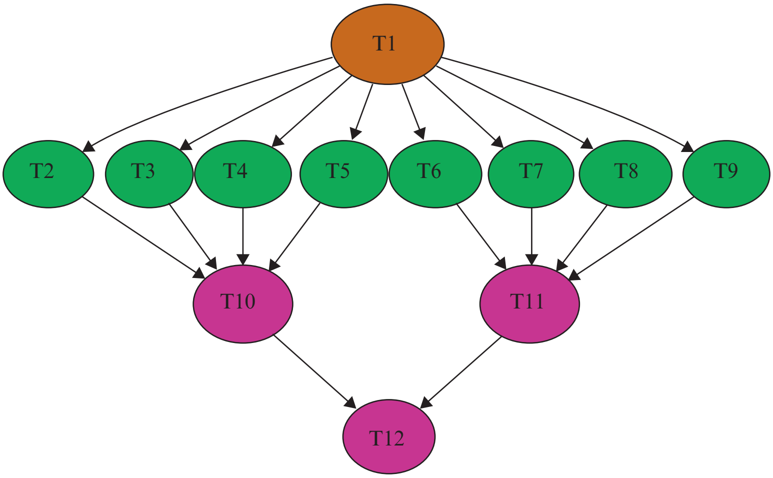


Fig. 4.1 Workflow

Workflow Multiplicity Algorithm

These algorithms are designed to schedule a single workflow instance, multiple instances of a same workflow or multiple workflows. Based on this there exist three types of scheduling processes from the perspective of workflow multiplicity.

Single Workflow

It is the traditional model offered for grid, cluster, and cloud. Here the algorithms are designed to optimize the single workflow schedule. Scheduler manages the workflow execution. Algorithm mainly focuses on cost optimization and other QoS requirements for single user and single DAG.

Workflow Ensembles

Many workflow applications are composed of one or more workflow instance. These workflows are interrelated and are grouped together to produce a desired result. Workflow ensembles have a similar structure with different sizes and input data. Scheduling algorithms for this type of workflow focus on executing every workflow in an ensemble with the available resources. Scheduling policies need to be taken care of satisfying the QoS requirements of all the workflows in an ensemble. An important characteristic of the ensemble is the number of instances available which is known in advance so that the scheduling policy may make use of it during the planning phase of the workflow execution.

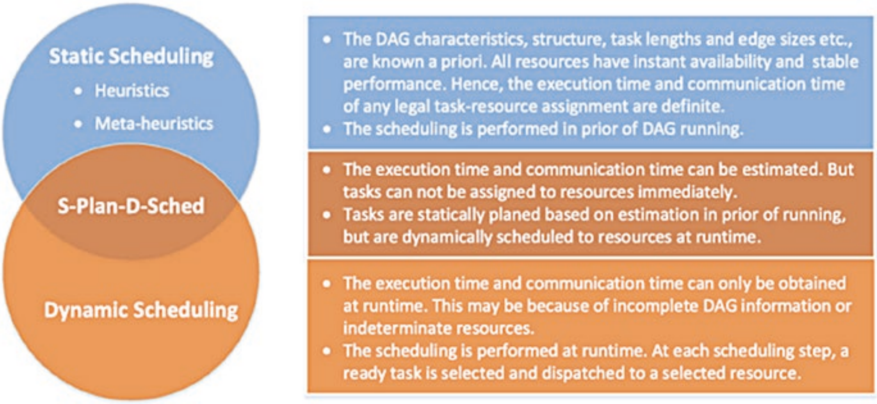
Multiple Workflows

Multiple workflows consist of unrelated workflows that vary in structure, size, and inputs. Number and types of workflows are not known in advance; hence the scheduling of this type of workflow is considered as a dynamic scheduling whose workloads are change dynamically with time. Each instance of workflow has its own independent QoS requirements. Scheduling policy for this type of workflows needs to be taken care of, scheduling the available resources efficiently to meet the QoS requirements of all the workflows.

Taxonomy of Workflow Scheduling

To differentiate a numerous workflow scheduling methods, Fuhai et al. classified them into three categories: static scheduling, S-plan-D-Sched, and dynamic scheduling based on the information of workflow, resource, and task assignment to resources [12]. The taxonomy is shown in the Fig. 4.2.

Comparative study shows that static scheduling algorithm outperforms dynamic scheduling in several cases. The reason is static scheduling searches a solution space globally at a workflow level and task level. It assumes the task execution, and communication time can be obtained earlier which is not always true in real systems. Both the resource and workflow information determine task execution and communication time. Workflow information includes structure of the workflow, task execution workload, and communication data size. The resource information includes availability, processing capability, communication bandwidth, etc. The performance fluctuation of multi-tenant resource sharing, hardware, and software failures is also taken into consideration in cloud computing environment due to its dynamic resource provisioning characteristics. Dynamic scheduling is able to



- The DAG characteristics, structure, task lengths and edge sizes etc., are known a priori. All resources have instant availability and stable performance. Hence, the execution time and communication time of any legal task-resource assignment are definite.
- The scheduling is performed in prior of DAG running.

- The execution time and communication time can be estimated. But tasks can not be assigned to resources immediately.
- Tasks are statically planed based on estimation in prior of running, but are dynamically scheduled to resources at runtime.

- The execution time and communication time can only be obtained at runtime. This may be because of incomplete DAG information or indeterminate resources.
- The scheduling is performed at runtime. At each scheduling step, a ready task is selected and dispatched to a selected resource.

Fig. 4.2 Taxonomy of workflow scheduling

handle these uncertainties. The integrated approach is called S-Plan-D-Sched which takes the advantage of both static and dynamic scheduling. It makes a static plan for all the tasks approximately based on the estimation of task execution and communication time. At runtime, the task assignments are adaptively tuned, and if necessary they are rescheduled.

4.3 Static Workflow Scheduling

Static scheduling assumes the task timing information precisely but incurs less runtime overhead. An example for a static scheduling algorithm is Opportunistic Load Balancing (OLB).

List Scheduling Heuristic
It creates a scheduling list by assigning priority and sorting the task according to their priority, and then repeatedly it selects the task and resource till all tasks in the DAG are scheduled. A prioritizing attribute and resource selection strategy are required to decide task priorities and optimal resource for each task.

Some list scheduling heuristics are modified critical path (MCP), mapping heuristic (MH), insertion scheduling heuristic and earliest time first (ETF), heterogeneous earliest finish time (HEFT), critical path on a processor (CPOP), dynamic level scheduling (DLS), dynamic critical path (DCP), and predict earliest finish time (PEFT).

Clustering Heuristic
Clustering heuristic is designed to optimize transmission time between data-dependent tasks. The two main parts of clustering heuristics are clustering and ordering. Clustering maps tasks to clusters, whereas ordering orders task in the same cluster.

Duplication Heuristic

Duplication technique is usually used along with list scheduling or clustering scheduling as an optimization procedure or as a new algorithm. There are two issues to be addressed when designing an effective task duplication algorithm:

1. Which task(s) to duplicate? The start time of the child task can be minimized by the selecting which parent talks to be duplicated.
2. Where to duplicate the task(s)? Allocating the proper time slot on the resources to the duplicate the parent task(s).

According to the selection of duplicate task, duplication algorithm is classified as scheduling with full duplication (SFD) and scheduling with partial duplication (SPD). Task from high priority or higher levels is considered for duplication in SFD.

Meta-heuristic

To achieve better optimization solutions, meta-heuristic approaches are used. Large complicated problems are solved by using this approach.

DAG scheduling is an NP-complete problem. Therefore, developing an approximate algorithm is a good alternative comparing to the exact methods. Some meta-heuristic solutions were proposed as they provide an efficient way toward a good solution. Genetic algorithm is one of the best solutions for task scheduling problem. Some of the examples for DAG scheduling includes [8, 11, 13, 14]. These genetic algorithms differ in string representations of schedules in search space, fitness function evaluation for schedules, genetic operators for generating new schedules, and stochastic assignment to control the genetic operators.

Blythe et al. [15] investigated greedy GRAP (Greedy Randomized Adaptive Search) algorithm for workflow scheduling on grids, which performs better than Min-Min heuristic for data-intensive applications. Young et al. [16] have investigated performance of simulated annealing (SA) algorithms for scheduling workflow applications in grid environment.

4.4 Dynamic Workflow Scheduling

Dynamic scheduling doesn't know about the task arrival information during runtime although it adapts better to the timing changes during execution. Some of the example of dynamic scheduling is Earliest Deadline First (EDF) and Least Laxity First (LLF). Dynamic scheduling is developed for handling the unavailability of scheduling information and resource contention with other workflow or non-workflow system load. Sonmez et al. [52] presented a dynamic scheduling taxonomy based on the three-resource information status, processing speed and link speed, and two-task information task length and communication data size. A dynamic scheduling algorithm balances the load among the available resource queues. Xie and Qin [17, 18] addressed a family of dynamic security-aware scheduling algorithms for homogeneous clusters and heterogeneous distributed systems.

4.5 Workflow Scheduling

Workflow scheduling is the problem of mapping of the workflow flow tasks on suitable resources while satisfying the constraints imposed by the users [19]. In other words, it is the process of atomization of the workflows with the help of algorithms. A workflow will consist of sequence of connected instructions. The motive of workflow scheduling is to automate the procedures especially which are involved in the process of passing the data and fields between the participants of the cloud maintaining the constraints [20]. The performance of the entire system can be achieved by properly scheduling the workflows with the help of the scheduling algorithms.

The WfMC (Workflow Management Coalition) defined workflow as "The automation of a business process, in whole a set of procedural rules" [21]. The components of the workflow reference model are represented in Fig. 4.3.

Workflow Engine
The workflow engine will provide a runtime environment to create, manage, and execute the workflow instances.

Process Definition
The processes are defined in such a way that it facilitates the automated manipulation.

Workflow Interoperability
Interoperability is provided between the different kinds of workflow systems.

Invoked Application
It helps in the communication between the different kinds of IT applications.

Workflow Client Application
Support for the interaction with the users with the help of user interface.

Administration and Monitoring
It helps in coordinating the composite workflow application environment.

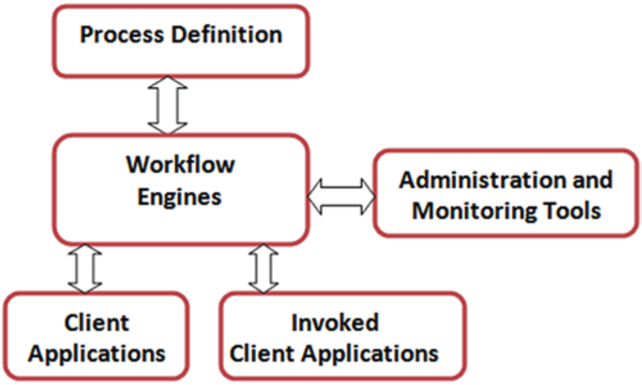

Process Definition
Workflow Engines
Administration and Monitoring Tools
Client Applications
Invoked Client Applications

Fig. 4.3 Workflow reference model [22]

4.6 Taxonomy of Cloud Resource Scheduling

Scheduling can be done in different layers of service stacks; hence the cloud computing architecture consists of IaaS, PaaS, and SaaS stacks which classifies the scheduling problem according to the stacks such as scheduling in application (software), scheduling in the virtualization (platform), and scheduling in the deployment (infrastructure) [23]. Figure 4.4 represents the taxonomy of cloud resource scheduling.

- Scheduling in the application layer schedules the virtual or physical resources to support for user applications, tasks, and workflows with optimal QOS and efficiency and software.
- Scheduling in the virtualization layer maps virtual resources to physical resources with load balance, energy efficiency, budget, and deadline constraints.
- Scheduling in the deployment layer is concerned with the infrastructure, service placement, multi-cloud centers, outsourcing, partnering, data routing, and application migration.

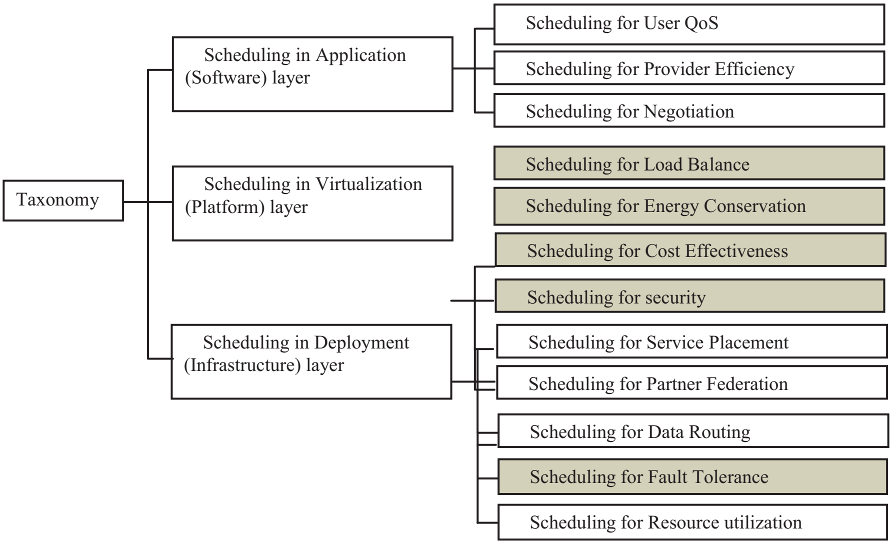

Taxonomy
Scheduling in Application (Software) layer
Scheduling in Virtualization (Platform) layer
Scheduling in Deployment (Infrastructure) layer
Scheduling for User QoS
Scheduling for Provider Efficiency
Scheduling for Negotiation
Scheduling for Load Balance
Scheduling for Energy Conservation
Scheduling for Cost Effectiveness
Scheduling for security
Scheduling for Service Placement
Scheduling for Partner Federation
Scheduling for Data Routing
Scheduling for Fault Tolerance
Scheduling for Resource utilization

Fig. 4.4 Taxonomy of cloud resource scheduling

4.7 Existing Workflow Scheduling Algorithms

To schedule workflow in a cloud environment, schedulers may consider different objectives and parameters such as execution time, cost, load balance, etc. as listed below. According to the objectives considered, the categorized scheduling algorithms are as follows:

4.7.1 Best Effort Workflow Scheduling

Best effort scheduling tries to optimize one objective leaving the other factors such as various QoS requirements. Pandey et al. [11, 24] presented a particle swam optimization-based heuristic to schedule workflows to cloud resources that target to minimize the cost. Both the execution cost and data transmission cost are taken into account. Simulation is performed in a cloud environment with the help of Amazon EC2. For algorithm performance, PSO is compared with best resource selection (BRS), and PSO achieves three times cost saving than BRS.

Bittencourt et al. [12, 25] presented Hybrid Cloud Optimized Cost schedule algorithm for scheduling workflow in a hybrid environment with an aim to optimize cost. It decides which resources should be leased from the public cloud and aggregated to the private cloud to provide sufficient processing power to execute a workflow within a given execution time.

Garg et al. [8, 26] proposed adaptive workflow scheduling(AWS) with an aim to minimize the makespan. It also considers the resource availability changes and the impact of existing loads over grid resources. Scheduling policy changes dynamically as per previous and current behavior of the system. Simulation is performed in grid environment with a help of GridSim toolkit. To verify the correctness of the algorithm, it is compared with a popular heuristics such as HEFT, Min-Min, Max-Min, AHEFT, Re-DCP-G, and Re-LSDS, and the result shows that AWS performed 10–40% better than the other scheduling algorithms considered.

Luo et al. [11, 27] presented a deadline guarantee enhanced scheduling algorithm (DGESA) for deadline scientific workflow in a hybrid grid and cloud environment that targets the deadline guarantee with the objective to minimize the makespan. The simulation environment is hybrid in nature with three grid sites and four cloud services. For the result analysis, DGESA is compared with HCOC and Aneka algorithm, and the result shows that the DGESA is very efficient in deadline guarantee and the makespan is lower than other two algorithms.

4.7.2 *Bi-objective Workflow Scheduling*

Verma et al. [13, 28] proposed BDHEFT which considers budget and deadline constraints while scheduling workflow with an aim to optimize the makespan and cost. Simulation is performed in cloud environment with a help of CloudSim. To evaluate the algorithm, authors compared BDHEFT with HEFT, and result shows that BDHEFT outperforms HEFT in terms of monetary cost and produces better makespan.

Udomkasemsub et al. [14, 29] proposed a workflow scheduling framework using Artificial Bee Colony (ABC) by considering multiple objective such as makespan and cost. A Pareto analysis concept is applied to balance the solution quality according to both objectives. Experiment is performed in cloud environment with java implementation. ABC outperforms HEFT/LOSS in both single objective optimization with constraints and multiple objective optimizations for structured workflow.

Wu et al. [15, 30] proposed Revised Discrete Particle Swarm Optimization (RDPSO) to schedule workflow applications in the cloud that takes both transmission cost and communication cost into account, which aims to minimize the makespan and cost of the workflow application. Experiments are performed in Amazon Elastic Compute Cloud. Experimental results show that the proposed RDPSO algorithm can achieve much more cost savings and better performance on makespan and cost optimization.

Ke et al. [16, 31] presented compromised-time-cost (CTC) scheduling algorithm with an aim to minimize the makespan and cost of the workflows. Simulation is performed in SwinDeW-C platform. For the performance evaluation, CTC is compared with Deadline-MDP algorithm, which shows that CTC performance is better in terms of cost of 15% and execution cost by 20%.

Arabnejad et al. [17, 32] proposed a Heterogeneous Budget-Constrained Scheduling (HBCS) algorithm that guarantees an execution cost within the user's specified budget and minimizes the makespan. The experiment is performed in GridSim toolkit, and for the evaluation of algorithm performance, it is compared with LOSS, GreedyTime-CD, and BHEFT. The result shows that HBCS algorithm achieves lower makespans, with a guaranteed cost per application and with a lower time complexity than other budget-constrained state-of-the-art algorithms.

Singh et al. [18, 33] proposed a score-based deadline constrained workflow scheduling algorithm, which considers deadline as the main constraint to minimize the cost while meeting user-defined deadline constraints. Simulation is performed in a cloud environment with a help of CloudSim toolkit. For performance analysis, the algorithm is compared with the same algorithm without score result that shows that score-based algorithm performance is better.

Verma et al. [19, 34] proposed Deadline and Budget Distribution-based Cost-Time Optimization (DBD-CTO) with an aim to minimize cost and time by considering the budget.

Malawski et al. [20, 35] addressed the problem of managing workflow ensembles with a goal to maximize the completion of user-prioritized workflows in a given budget and deadline. They developed Dynamic Provisioning Dynamic Scheduling (DPDS), workflow-aware DPDS, and Static Provisioning Static Scheduling (SPSS) to solve the problem. The performance of the algorithms is evaluated using workflow ensembles with a consideration of budget, deadline, task runtime estimations, failures, and delays. The result shows that the algorithm WA-DPDS and DPDS performed better than the SPSS.

Xu et al. [21, 36] introduced a Multiple QoS Constrained Scheduling Strategy of Multi-Workflows (MQMW) with an aim to minimize the execution time and cost and increase the success rate for Multi-Workflows. They evaluated the performance of the algorithm in the experimental simulator with a randomly generated workflow graph and compared with RANK-HYBD. The result shows MQMW performed better than RANK-HYBD in all aspects.

Bessai et al. [22, 37] proposed three complementary bi-criteria approaches for scheduling workflows in a cloud environment with an aim to minimize time and cost. The first approach focused on the cost of utilizing the set of resources, while the second approach tries to minimize the overall execution time, and the third approach is called cost-time approach, which is based on the Pareto solution of first two approaches.

Chopra et al. [23] presented a HEFT-based hybrid scheduling algorithm with a concept of sub-deadline for rescheduling and resource allocation for scheduling workflow application in the hybrid cloud. The algorithm finds the best resource from public cloud within cost and deadline. For performance analysis, the proposed algorithm is compared with Greedy and Min-Min approaches. The result shows that the proposed algorithm performed better in terms of cost and completed all tasks within the given budget.

Verma et al. [24] had Extended Biobjective Priority-Based Particle Swarm Optimization (BPSO) algorithm with an aim to minimize the cost under deadline and budget constraints. Simulation is performed in CloudSim and compared with modified BTGA and BHEFT. The result shows that Extended BPSO outperforms both algorithms.

Lin et al. [25, 40] proposed the SHEFT workflow scheduling algorithm to schedule workflows elastically on a cloud environment. It provides the resource scalability according to the needs and provides better execution time.

Poola et al. [26, 41] presented a just-in-time and adaptive scheduling heuristic that maps workflow tasks onto the spot and on-demand instance with an aim to minimize the execution cost and also provides a robust scheduling that satisfies the deadline constraint.

4.7.3 Multi-objective Workflow Scheduling

Multiple objective workflow scheduling considers several objectives, in order to make an efficient scheduling. Bilgaiyan et al. [27, 42] presented a cat swarm-based multi-objective optimization for scheduling workflows in a cloud environment.

Main objectives considered are cost, makespan, and CPU idle time. Experiments are performed in a MATLAB tool, and the comparison is made with the existing multi-objective particle swarm optimization (MOPSO); the result shows that multi-objective cat swarm optimization performance is better than MOPSO.

Kumar et al. [28, 43] proposed apriority-based decisive algorithm with De-De dodging algorithm that is proposed to schedule multiple workflows with an aim to maximize the resource utilization and cost and reduce the makespan. CloudSim was used for the simulation and the proposed algorithm is compared with the Time and Cost Optimization for Hybrid Clouds (TCHC) algorithm and De-De, PBD performs better than TCHC. The De-De algorithms performs better in terms of CPU time, makespan, and cost.

Yassa et al. [29, 44] proposed Dynamic Voltage and Frequency Scaling (DVFS)-Multi-Objective Discrete Particle Swarm Optimization (DVFS-MODPSO) to minimize makespan, cost, and energy. Dynamic Voltage and Frequency Scaling technique is used to reduce the consumption of energy. The proposed algorithm is compared with the HEFT for performance, and the result shows DVFS-MODPSO performance is better and produces an optimal solution.

Li et al. [30, 45] presented a security and cost-aware scheduling (SCAS) with an aim to minimize the cost by considering deadline and risk rate constraints. They deployed security services such authentication service, integrity service, and confidentiality service to protect the workflow applications from common security attacks. CloudSim is used for the simulation and experiment is conducted with the scientific workflows.

Jayadivya et al. [31, 46] proposed a QWS algorithm with an aim to achieve minimum makespan and cost and maximum reliability. For the performance evaluation, proposed QWS algorithm is compared with MQMW algorithm, and the result shows that the success rate of the QWS algorithm is better than MQMW.

The detailed comparison of cloud scheduling algorithms is depicted in Tables 4.1 and 4.2. The algorithms are classified based on best effort scheduling and bi-objective and multi-objective scheduling. The type of workflow and environment in which these algorithms are experimented or simulated is compared.

From the comparison it is clear that the major scheduling algorithm considers the objective makespan, deadline, and budget, and there are very less number of works toward the other objectives. There is a need to be taken care of other objectives to make an efficient and effective workflow scheduling. Rahman et al. [32, 47] presented dynamic critical path-based workflow scheduling algorithm for the grid (DCP-G) that provides an efficient schedule in astatic environment. It adapts to dynamic grid environment where resource information is updated after fixed interval, and rescheduling (Re-DCP-G) will be done if necessary. A. Olteanu et al. [33, 48] proposed a generic rescheduling algorithm that will be useful for large-scale distributed systems to support fault tolerance and resilience. As energy consumption is becoming an important factor, it also becomes an issue for scheduling workflows. Two types of solution are available to attain the target of energy consumption reduction [1, 34]. They are resource utilization and dynamic voltage scaling (DVS). Reduction of energy consumption is done with the help of improving the resource

Table 4.1 Comparison of cloud scheduling algorithms

Algorithm	Type of workflow	Environment	Simulation environment
Best effort workflow scheduling			
Pandey et al. [24]	Scientific workflow	Cloud	Jswarm
Bittencourt et al. [25]	Scientific workflow	Hybrid cloud	Amazon EC2
Garg et al. [26]	Simple workflow	Grid	GridSim
Luo et al. [27]	Scientific workflow	Hybrid (grid and cloud)	–
Bi-objective workflow scheduling			
Verma et al. [28]	Scientific workflow	Cloud	CloudSim
Udomkasemsub et al. [29]	Scientific workflow	Cloud	Java
Wu et al. [30]	Simple workflow	Cloud	Amazon EC2
Ke et al. [31]	Simple workflow	Cloud	SwinDew
Arabnejad et al. [32]	Scientific workflow	Grid	GridSim
Singh et al. [33]	Simple workflow	Cloud	CloudSim
Verma et al. [34]	Simple workflow	Cloud	Java
Malawski [35]	Multiple workflow	Cloud	Cloud workflow simulator
Xu et al. [36]	Simple workflow	Cloud	–
Bessai et al. [37]	Scientific workflow	Cloud	–
Chopra et al. [38]	-	Hybrid cloud	–
Verma et al. [39]	Scientific workflow	Cloud	CloudSim
Lin et al. [40]	Scientific workflow	Cloud	–
Poola et al. [41]	Scientific workflow	Cloud	–
Multi-objective workflow scheduling			
Bilgayan et al. [42]	Simple workflow	Cloud	MATLAB
Kumar et al. [43]	Simple workflow	Cloud	CloudSim
Yassa et al. [44]	Scientific workflow	Cloud	–
Li et al. [45]	Scientific workflow	Cloud	CloudSim
Jayadivya et al. [46]	Multiple workflow	Cloud	–

Table 4.2 Comparison of objectives of cloud scheduling algorithms

Algorithm	Makespan	Deadline	Budget	Security	Energy	Elasticity	Reliability	Resource utilization
Best effort workflow scheduling								
Pandey et al. [24]	–	–	✓	–	–	–	–	–
Bittencourt et al. [25]	–	–	✓	–	–	–	–	–
Garg et al. [26]	✓	–	–	–	–	–	–	–
Luo et al. [27]	✓	–	–	–	–	–	–	–
Bi-objective workflow scheduling								
Verma et al. [28]	✓	–	✓	–	–	–	–	–
Udomkasemsub et al. [29]	✓	–	✓	–	–	–	–	–
Wu et al. [30]	✓	–	✓	–	–	–	–	–
Ke et al. [31]	✓	–	✓	–	–	–	–	–
Arabnejad et al. [32]	✓	–	✓	–	–	–	–	–
Singh et al. [33]	–	✓	✓	–	–	–	–	–
Verma et al. [34]	–	✓	✓	–	–	–	–	–
Malawski [35]	–	✓	✓	–	–	–	–	–
Xu et al. [36]	✓	–	✓	–	–	–	–	–
Bessai et al. [37]	–	✓	✓	–	–	–	–	–
Chopra et al. [38]	–	✓	✓	–	–	–	–	–
Verma et al. [39]	–	✓	✓	–	–	–	–	–
Lin et al. [40]	✓	–	–	–	–	✓	–	–
Poola et al. [41]	–	✓	✓	–	–	–	–	–
Multi-objective workflow scheduling								
Bilgayan et al. [42]	✓	–	✓	–	–	–	–	✓
Kumar et al. [43]	✓	–	✓	–	–	–	–	✓
Yassa et al. [44]	✓	–	✓	–	✓	–	–	–
Liet al. [45]	–	✓	✓	✓	–	–	–	–
Jayadivya et al. [46]	–	✓	✓	–	–	–	✓	–

utilization [35–37, 49–51]. Venkatachalam et al. [38, 52] investigated and developed various techniques such as DVS, memory optimizations, and resource hibernation. DVS has been proven an efficient technique for energy savings [39, 40, 53, 54].

For workflow scheduling problem in the cloud, the best method is to generate a schedule using makespan best effort algorithm and then adopt DVS technique to tune the slack time of the generated schedule [41, 42, 55, 56]. The removal of data files that are no longer needed is a best practice toward an efficient mapping and execution of workflows, since it minimizes their overall storage requirement. Piotr bryk et al. [43] proposed and implemented a simulation Model for handling file transfers dynamically between the tasks with configurable replications. Also, they proposed Dynamic Provisioning Locality-Aware Scheduling (DPLS) and Storage- and Workflow-Aware DPLS (SWA-DPLS) algorithm. Yang Wang et al. [44, 57] presented WaFS, a user-level workflow-aware file system with a proposed hybrid scheduling (HS) algorithm for scientific workflow computation in the cloud. Workflow scheduler usesWaFS data to make effective cost-performance trade-offs or improve storage utilization.

4.8 Issues of Scheduling Workflow in Cloud

The IaaS and PaaS combination cloud forms a complete workflow scheduling architecture, and it introduce new challenges [34]:

- Scheduling works for grids and clusters focused on meeting deadlines or minimizing makespan without considering the cost. New scheduling algorithms need to be developed for the cloud by considering the pay-as-you-go model in order to avoid unnecessary cost.
- Adaptive nature: Cloud is a dynamic environment with a variation of performance during the workflow execution; therefore an adaptive scheduling solution is required.
- Migration: Migration of jobs for load balancing also affects the performance of workflow scheduling
- Virtual machine replacement: Workflow scheduling algorithm should efficiently replace the VM in the case of failure.
- There are two main stages in cloud environment prior to the execution of workflow: Resource provisioning and task-resource mapping. Grid and cluster environments focus only on task-resource mapping stage since they are static environment whose configurations are known in advance. A major issue of resource provisioning is to determine the amount and resource type that a workflow application would request, which affects the cost and makespan of a workflow.
- Virtual machines offered by current cloud infrastructure are not exhibiting stable performance [45]. This has a significant impact on a scheduling policy. VM boot

time and stop time are the other important factors that should be considered for cloud scheduling [46].

- Integrated architecture: The first challenge is to integrate the workflow management system with the cloud. A workflow engine should be designed with strong functionality to deal with large-scale tasks.
- Large-scale data management and workflow scheduling: Scientific workflow applications are more data intensive, and data resource management and data transfer between the storage and computing resources are the main bottleneck. It is very important to find an efficient way to manage data needed by the workflows.
- Service composition and orchestration: To accommodate the further growing and complex workflow application needs, services of several cloud providers should be composed to deliver uniform QoS as a single request. To achieve the consumer requirements, this composition and orchestration of services should be carried out in an automated and dynamic manner. To find an efficient composition solution in Intercloud is a major challenge as it involves service composition and orchestration optimization under the constraint of cost and deadline.

4.9 Conclusion

Cloud environment provides unprecedented scalability to workflow systems and changes the way we perceive and conduct experiments. Numerous researches have been conducted about the workflow scheduling with different objectives to obtain optimized solutions. This chapter describes analysis of various existing workflow scheduling algorithms, issues, and challenges of scheduling workflow in a cloud environment. The algorithms are classified based on the objectives and a comparison is made. Budget is the crucial issue that is addressed in most of the works, but other issues such as security, energy, scalability, and reliability have been less addressed; there should be a consideration in these objectives in order to make an efficient schedule. And also there is a less number of works toward the multi-cloud environment as the complexity of workflows is getting increased a single cloud cannot satisfy the requirement. Most of the existing algorithms are suitable only for single cloud, and there is a need for running workflow management system on multi-cloud; therefore, there is a wide opportunity to develop scheduling algorithm that satisfies the multi-cloud properties.

References

1. Wu F, Wu Q, Tan Y (2015) Workflow scheduling in cloud: a survey. J Supercomputing 71(9):3373–3418

2. Ullman JD (1975) Np-complete scheduling problems. J Comput Syst Sci 10(3):384–393. Math SciNetCrossRefMATHGoogle Scholar
3. Mell P, Grance T (2009) The nist definition of cloud computing. Natl Inst Stand Technol 53(6):50Google Scholar
4. Amazon ec2 pricing. http://aws.amazon.com/ec2/pricing/
5. Abawajy JH (2004) Fault-tolerant scheduling policy for grid computing systems. In: Proceedings of parallel and distributed processing symposium, 2004, 18th international, IEEE, p 238
6. Abrishami S, Naghibzadeh M, Epema DH (2012) Cost-driven scheduling of grid workflows using partial critical paths. IEEE Trans Parallel Distrib Syst 23(8):1400–1414
7. Abrishami S, Naghibzadeh M, Epema DH (2013) Deadline-constrained workflow scheduling algorithms for infrastructure as a service clouds. Futur Gener Comput Syst 29(1):158–169
8. Ahmad I, Dhodhi MK (1995) Task assignment using a problem genetic algorithm. Concurr Pract Exp 7(5):411–428
9. Ahmad I, Kwok YK (1998) On exploiting task duplication in parallel program scheduling. IEEE Trans Parallel Distrib Syst 9(9):872–892
10. Ali S, Maciejewski AA, Siegel HJ, Kim JK (2004) Measuring the robustness of a resource allocation. IEEE Trans Parallel Distrib Syst 15(7):630–641
11. Ali S, Sait SM, Benten MS (1994) Gsa: scheduling and allocation using genetic algorithm. In: Proceedings of the conference on European design automation, IEEE, pp 84–89
12. Mohanapriya N, Kousalya G, Balakrishnan P, Cloud workflow scheduling algorithms: a survey. Int J Adv Eng Technol E-ISSN 0976-3945
13. Hou ESH, Ansari N, Ren H (1994) A genetic algorithm for multiprocessor scheduling. IEEE Trans Parallel Distrib Syst 5(2):113–120
14. Wu AS, Yu H, Jin S, Lin KC, Schiavone G (2004) An incremental genetic algorithm approach to multiprocessor scheduling. IEEE Trans Parallel Distrib Syst 15(9):824–834
15. Blythe J, Jain S, Deelman E, Gil Y, Vahi K, Mandal A, Kennedy K (2005) Task scheduling strategies for workflow-based applications in grids. In: Proceedings of cluster computing and the grid, CCGrid 2005, vol 2, IEEE International Symposium on 2005, pp 759–767
16. Young L, McGough S, Newhouse S, Darlington J (2003) Scheduling architecture and algorithms within the iceni grid middleware. In: Proceedings of UK e-science all hands meeting, Citeseer, pp 5–12
17. Xie T, Qin X (2006) Scheduling security-critical real-time applications on clusters. IEEE Trans Comput 55(7):864–879
18. Xie X, Qin (2007) Performance evaluation of a new scheduling algorithm for distributed systems with security heterogeneity. J Parallel Distrib Comput 67(10):1067–1081
19. Kataria D, Kumar S (2015) A study on workflow scheduling algorithms in cloud. Int J Res Appl Sci Technol 3(8)
20. Tao J, Kunze M, Rattu D, Castellanos AC (2008) The cumulus project: build a scientific cloud for a data center. In: Cloud Computing and its Applications, Chicago
21. Chopra N, Singh S, Survey on scheduling in hybrid clouds, 5th ICCCNT–2014 July 1113, 2014, Hefei, China
22. Yadav SS, Hua ZW (2010) CLOUD: a computing infrastructure on demand, IEEE
23. Chopra N, Singh S (2013) HEFT based workflow scheduling algorithm for cost optimization within deadline in hybrid clouds. In: Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference, pp 1–6. doi:10.1109/ICCCNT.2013.6726627
24. Pandey S, Wu L, Guru SM, Buyya R A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments. In: Advanced Information Networking and Applications (AINA), 2010 24th IEEE international conference, pp 400–407. 20–23 April 2010. doi:10.1109/AINA.2010.31

25. Bittencourt LF, Madeira ERM HCOC: a cost optimization algorithm for workflow scheduling in hybrid clouds. Int J Internet Serv Appl 2:207–227. doi:10.1007/s13174-011-0032-0
26. Garg R, Singh AK (2015) Adaptive workflow scheduling in grid computing based on dynamic resource availability. Eng Sci Technol An Int J 18(2):256–269. http://dx.doi.org/10.1016/j.jestch.2015.01.001
27. Luo H, Yan C, Hu Z (2015) An Enhanced Workflow Scheduling Strategy for Deadline Guarantee on Hybrid Grid/Cloud Infrastructure. J Appl Sci Eng 18(1):6778. doi:10.6180/jase.2015.18.1.09
28. Verma A, Kaushal S (2015) Cost-time efficient scheduling plan for executing workflows in the cloud. J Grid Comput 13(4):495–506. doi:10.1007/s10723-015-9344-9
29. Udomkasemsub O, Xiaorong L, Achalakul T (2012) A multiple-objective workflow scheduling framework for cloud data analytics. In: Computer Science and Software Engineering (JCSSE), International Joint Conference, pp 391–398. doi:10.1109/JCSSE.2012.6261985
30. Wu Z, Ni Z, Gu L, Liu X A revised discrete particle swarm optimization for cloud workflow scheduling. In: Computational Intelligence and Security (CIS), 2010 International conference on 11–14 Dec 2010, pp 184–188. doi:10.1109/CIS.2010.46
31. Ke L, Hai J, Jinjun CXL, Dong Y (2010) A compromised time-cost scheduling algorithm in SwinDeW-C for instance-intensive cost-constrained workflows on cloud computing platform. J High-Perform Comput Appl 24(4):445–456
32. Arabnejad H, Barbosa JG (2014) A budget constrained scheduling algorithm for workflow applications. J Grid Comput 12(4):665–679. http://dx.doi.org/10.1007/s10723-014-9294-7
33. Singh R, Singh S (2013) Score based deadline constrained workflow scheduling algorithm for cloud systems. Int J Cloud Comput Serv Archit 3(6). doi:10.5121/IJCCSA.2013.3603
34. Verma A, Kaushal S, Deadline and budget distribution based cost- time optimization workflow scheduling algorithm for cloud. In: IJCA Proceedings on International Conference on Recent Advances and Future Trends in Information Technology (iRAFIT 2012) iRAFIT(7):1–4
35. Malawski M, Juve G, Deelman E, Nabrzyski J, Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds. In: Proceedings of the international conference on high-performance computing, networking, storage and analysis (SC '12). IEEE Computer Society Press, Los Alamitos, CA, USA, Article 22
36. Xu M, Cui L, Wang H, Bi Y (2009) A multiple QoS constrained scheduling strategy of multiple workflows for cloud computing. In: Parallel and distributed processing with applications, 2009 IEEE International Symposium on, Chengdu, pp 629–634. doi: 10.1109/ISPA.2009.95
37. Bessai K, Youcef S, Oulamara A, Godart C, Nurcan S (2012) Bi-criteria workflow tasks allocation and scheduling in cloud computing environments. In: Cloud Computing (CLOUD), IEEE 5th International Conference, pp 638–645. doi: 10.1109/CLOUD.2012.83
38. Chopra N, Singh S (2013) HEFT based workflow scheduling algorithm for cost optimization within deadline in hybrid clouds. In: Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference, pp 1–6. doi: 10.1109/ICCCNT.2013.6726627
39. Verma A, Kaushal S (2015) Cost minimized PSO based workflow scheduling plan for cloud computing. IJITCS 7(8):37–43. doi:10.5815/ijitcs.2015.08.06
40. Lin C, Lu S (2011) Scheduling scientific workflows elastically for cloud computing. In: Cloud computing 2011 IEEE, international conference, Washington, DC, pp 746–747. doi: 10.1109/CLOUD.2011.110
41. Poola D, Ramamohanarao K, Buyya R (2014) Fault-tolerant workflow scheduling using spot instances on clouds. Proc Comput Sci 29:523–533
42. Bilgaiyan S, Sagnika S, Das M (2014) A multi-objective cat swarm optimization algorithm for workflow scheduling in cloud computing environment. Intell Comput Commun Devices 308:73–84. http://dx.doi.org/10.1007/978-81-322-2012-1_9
43. Kumar B, Ravichandran T Scheduling multiple workflow using De-De Dodging Algorithm and PBD Algorithm in cloud: detailed study. Int J Comput Electr Autom Control and Inf Eng 9(4):917–922

44. Yassa S, Chelouah R, Kadima H, Granado B (2013) Multi-objective approach for energy-aware workflow scheduling in cloud computing environments. Sci World J:Article ID 350934. doi:10.1155/2013/350934
45. Li Z, Ge J, Yang H, Huang L, Hu H, Hu H, Luo B (2016) A security and cost aware scheduling algorithm for heterogeneous tasks of scientific workflow in clouds. J Futur Gener Comput Syst. http://dx.doi.org/10.1016/j.future.2015.12.014
46. Jayadivya S, Bhanu SMS Qos based scheduling of workflows in cloud computing. Int J Comput Sci Electr Eng 2012:2315–4209
47. Rahman M, Hassan R, Ranjan R, Buyya R (2013) Adaptive workflow scheduling for dynamic grid and cloud computing environment. Concurr Comput Pract Exp 25(13):1816–1842. doi:10.1002/cpe.3003
48. Olteanu A, Pop F, Dobre C, Cristea V (2012) A dynamic rescheduling algorithm for resource management in large-scale dependable distributed systems. Comput Math Appl 63(9):1409–1423. http://dx.doi.org/10.1016/j.camwa.2012.02.066
49. Beloglazov A, Abawajy J, Buyya R Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Futur Gener Comput Syst 28(5):755–768. http://dx.doi.org/10.1016/j.future.2011.04.017
50. Beloglazov A, Buyya R (2012) Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers. Concurr Comput Pract Exp 24(13):1397–1420. http://dx.doi.org/10.1002/cpe.1867
51. Meng X, Pappas V, Zhang L (2010) Improving the scalability of data center networks with traffic-aware virtual machine placement. In: Proceedings of the 29th conference on information communications. IEEE Press, Piscataway, NJ, USA, pp 1154–1162
52. Sonmez O, Yigitbasi N, Abrishami S, Iosup A, Epema D (2010) Performance analysis of dynamic workflow scheduling in multicluster grids. In: Proceedings of the 19th ACM international symposium on high performance distributed computing, ACM, pp 49–60
53. Ge R, Feng X, Cameron KW, Performance-constrained distributed DVS scheduling for scientific applications on power-aware clusters. In: Supercomputing, 2005. Proceedings of the ACM/IEEE SC 2005 Conference, pp 34–34, 12–18 Nov 2005. doi: 10.1109/SC.2005.57
54. Rountree B, Lowenthal DK, Funk S, Freeh VW, de Supinski BR, Schulz M Bounding energy consumption in large-scale MPI programs. In: Supercomputing, 2007. SC '07. Proceedings of the 2007 ACM/IEEE Conference, pp 1–9, 10–16, Nov 2007. doi: 10.1145/1362622.1362688
55. Baskiyar S, Abdel-Kader R (2010) Energy-aware DAG scheduling on heterogeneous systems. Clust Comput 13(4):373–383. http://dx.doi.org/10.1007/s10586-009-0119-6
56. Cao F, Zhu MM, Wu CQ, Energy-efficient resource management for scientific workflows in clouds. In: Services, 2014 IEEE world congress, pp 402–409, July 2014 doi: 10.1109/SERVICES.2014.76
57. Wang Y, Lu P, Kent KB (2015) WaFS: a workflow-aware file system for effective storage utilization in the cloud. Comput IEEE Trans 64(9):2716–2729. doi:10.1109/TC.2014.2375195

Chapter 5
Workflow Modeling and Simulation Techniques

Abstract Modeling and simulation of scientific workflow play a vital role in resource allocation in a distributed environment. Simulation is one of the methods to solve the complex scientific workflows in distributed environment. There are many scientific workflow simulation software frameworks that are available for grid and cloud environment. WorkflowSim is an open-source simulator. WorkflowSim Simulator extends the existing CloudSim Simulator. The architecture, components, and scheduling algorithms used and also the simulation results are explained for CloudSim Simulator and WorkflowSim Simulator.

5.1 Introduction

Real-world process or system is imitated using simulation. The process of simulating a system requires a model. The model is developed, which depicts the fundamental properties or behavior or functions of the physical or abstract system or process. Here, the system is illustrated by the model, while the simulation portrays the operation of the system in due course of time.

Simulation is one of the popular valuation techniques in scientific workflow systems. Cloud computing environment supports dynamic provisioning of infrastructure for high-performance computing. WorkflowSim framework supports heterogeneous system overheads and failures. It is widely used in workflow management and optimization techniques; it extends the existing CloudSim. These two simulators provide unique and effective platforms for effective evaluation support for workflow management system. Also, it enables modeling and simulation of dynamic provisioning environment for cloud computing. Further it supports both system and behavior modeling for data centers, virtual machines, and dynamic resource provisioning.

CloudSim-based simulation provides dynamic support such as:

1. Creating necessary infrastructure as a service and testing them dynamically
2. Adaptive application testing for infrastructure provisioning
3. Performance tuning of interconnected and virtualized resources dynamically

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_5

CloudSim is a framework for modeling and simulation of cloud computing infrastructures and services. With the help of CloudSim, the following features can be achieved easily [1]:

- Modeling and simulation of large-scale IaaS
- Enables modeling as well as simulation of user-friendly policies for provisioning of resources and virtual machines
- Recently ContainerCloudSim support for containers (containers as a service) [2]
- Energy-aware computing services and resource utilization
- NetworkCloudSim for data center network topologies design and management [3]
- Supports user-specified policies for hosts to VM assignment as well as providing host resources to VM [1]

5.2 Architecture of CloudSim

CloudSim model of cloud computing architecture as shown in Fig. 5.1 [4] consists of the following three layers:

- The system layer
- The core middleware
- The user-level middleware

System Layer

The massive resources (RAM, storage, CPU, network) power the IaaS in the cloud computing environment. The resources are dynamically provisioned in the data centers with the help of virtualization techniques. The virtualized servers play a major role in this layer with necessary security and fault tolerance.

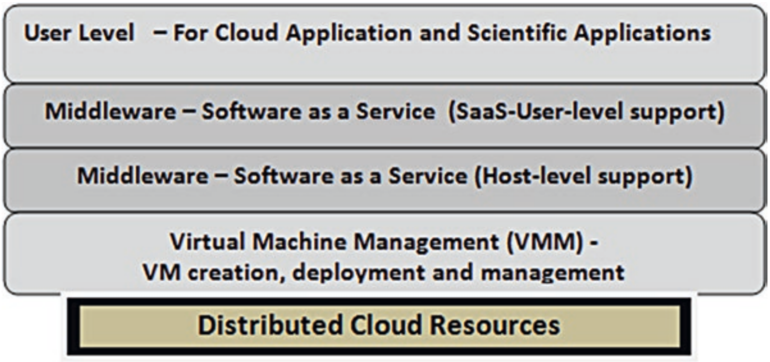

User Level – For Cloud Application and Scientific Applications
Middleware – Software as a Service (SaaS-User-level support)
Middleware – Software as a Service (Host-level support)
Virtual Machine Management (VMM) -
VM creation, deployment and management
Distributed Cloud Resources

Fig. 5.1 CloudSim model of cloud computing architecture [4]

Core Middleware

The virtualized resource deployment and management services are important service in this layer. The Platform as a Service (PaaS) is supported by the core middleware. It provides the following functionalities:

- Dynamic service-level agreement (SLA) management
- Accounting and metering services
- Execution monitoring and management
- Service discovery
- Load balancing

User-Level Middleware

This layer supports cost-effective user interface framework, programming environment, composition tools for creation, deployment, and execution of cloud applications.

Cloud Application

The various applications provided by the cloud service providers are supported in this cloud application layer as Software as a Service (Saas) for the end user.

5.3 Layered Design and Implementation of CloudSim Framework

Figure 5.2 demonstrates the layered architecture of the CloudSim framework. The very first layer in the CloudSim is the "user code" that creates the following things: multiple users along with their applications, VMs, and other entities available to the hosts. The next layer is CloudSim which is made up of many fundamental classes in java.

User Code

The user code is the first layer of simulation stack which expresses the configuration-related operations for hosts, applications, VMs, number of users, and their application types and broker scheduling policies.

CloudSim

The second layer, CloudSim, will consist of the following classes which are all the building blocks of the simulator.

The Data Center

- Data center class models the core infrastructure-level services (hardware and software) offered by resource providers in a cloud computing environment [5].
- It abbreviates a group of compute hosts which may be either homogeneous or heterogeneous with regard to their resource properties (memory, cores, capacity, and storage).

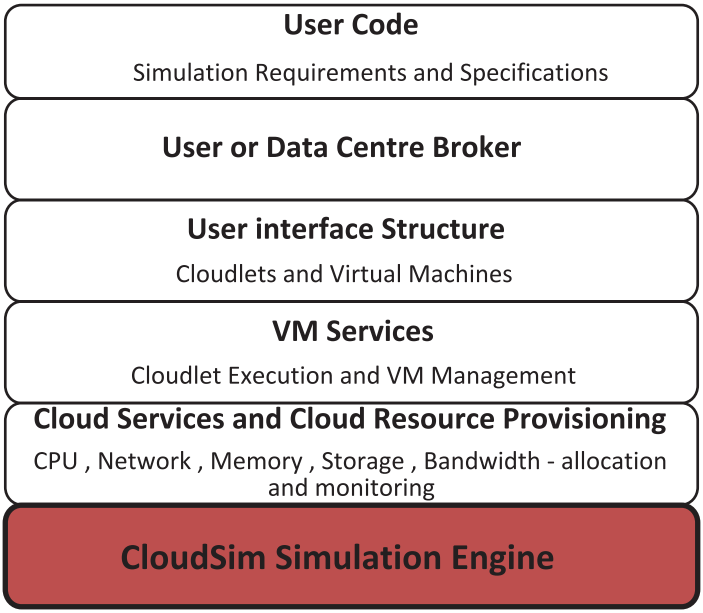

User Code
Simulation Requirements and Specifications
User or Data Centre Broker
User interface Structure
Cloudlets and Virtual Machines
VM Services
Cloudlet Execution and VM Management
Cloud Services and Cloud Resource Provisioning
CPU , Network , Memory , Storage , Bandwidth - allocation and monitoring
CloudSim Simulation Engine

Fig. 5.2 Layered implementation of the CloudSim framework [4]

The Data Center Broker

A broker is modeled by this class that is accountable for arbitrating between users and service providers based on users QoS demands and deploys service tasks across clouds.

SAN Storage

- This class defines a storage area network for cloud-based data centers to save huge shares of data.
- SAN storage realizes a simple interface through which storage and retrieval of any quantum of data at any time subject to the availability of network bandwidth [6].

Virtual Machine

- Virtual machine class realizes the instance of VM which is managed by the host component.
- Each VM has the capability to access to a component that has the characteristics related to a VM [7], such as memory, processor, storage, and the VM's internal scheduling policy, which is an expansion of abstract component named VM scheduling.

Cloudlet

Cloudlet class models the cloud-based application services which are commonly deployed in the data centers [8].

Cloud Coordinator

This abstract class enables the association capability to a data center. Further, it is responsible for intercloud coordinator communication and cloud coordinator and broker communication. Besides, it captures the inner state of a data center that plays a vital role in load balancing and application scaling.

BW Provisioner

BW provisioner is an abstract class for representing the bandwidth provisioning policy to VMs which are installed on a host entity.

Memory Provisioner

Memory provisioner is an abstract class for representing the memory provisioning policy to VMs. The VM is allowed to deploy on a host if and only if the memory provisioner identifies that the host has sufficient quantum of free memory as demanded by the new VM.

VM Provisioner

VM provisioner is an abstract class to define an allocation policy which is used by VM monitor to assign VMs to hosts.

VM Allocation Policy

VMM allocation policy is an abstract class which is realized by a host component that defines the policies required for allocating processing power to VMs [9].

Workflow Management System (WMS)

WMS is used for the management of the workflow tasks on the computing resources. The major components of the WMS are shown in Fig. 5.3.

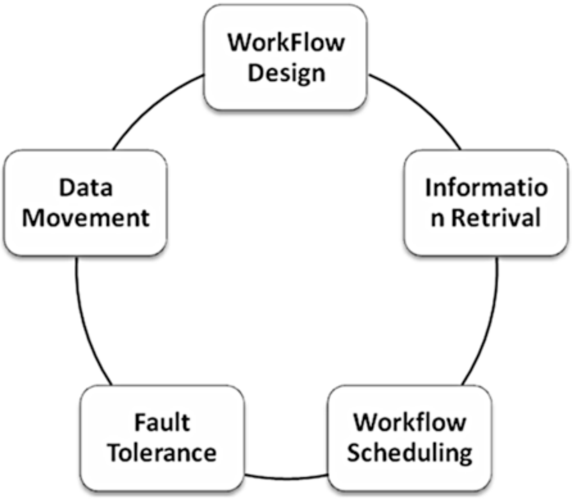


Fig. 5.3 Workflow structure

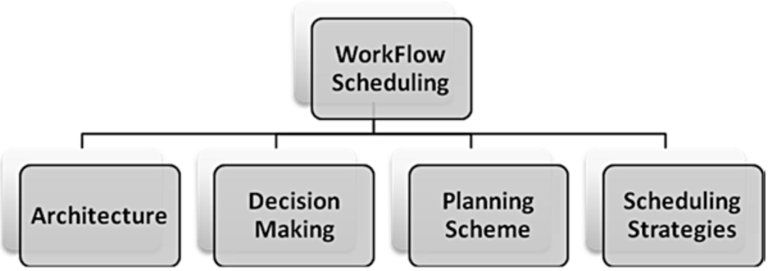


Fig. 5.4 Workflow scheduling

The workflow structure describes the relationship between the tasks of the workflow. It can be mainly of two types: DAG (Directed Acyclic Graph) and non-DAG. It can also be further categorized into sequence, parallelism, and choice in the DAG scheme.

Workflow Scheduling

Mapping and management of workflow task execution on shared resources are done with the help of workflow scheduling. The elements of the workflow scheduling are shown in Fig. 5.4.

Further subclassifications:

- Architecture:
  - Centralized
  - Hierarchical
  - Decentralized
- Decision-making:
  - Local
  - Global
- Planning scheme:
  - Static

    User-directed
    Simulation-based
  - Dynamic:

    Prediction-based
    Just in time
- Scheduling strategies:
  - Performance-driven
  - Market-driven
  - Trust-driven

5.4 Experimental Results Using CloudSim

Example 1: This example demonstrates the creation of two data centers with one host each and executes two cloudlets on them.

1. Initialize the CloudSim package.
2. Initialize the GridSim library.
3. Create the data centers:

   In CloudSim, the resources are provisioned from data centers. Further, there should be minimum one data center to run a CloudSim simulation.
4. Create the broker.
5. Create a VM by specifying its characteristics such as MIPS, size, RAM, and bandwidth.
6. Add those VMs into the VM list.
7. Submit VM list to the broker.
8. Create the cloudlets by specifying length in Million Instructions, input and output file size.
9. Add those cloudlets into cloudlet list.
10. Start the simulation.
11. In this example, the VMAllocatonPolicy in use is SpaceShared. It means that only one VM is allowed to run on each Pe (processing element). As each host has only one Pe, only one VM can run on each host.
12. Finally, the simulation results are printed.

Output Results

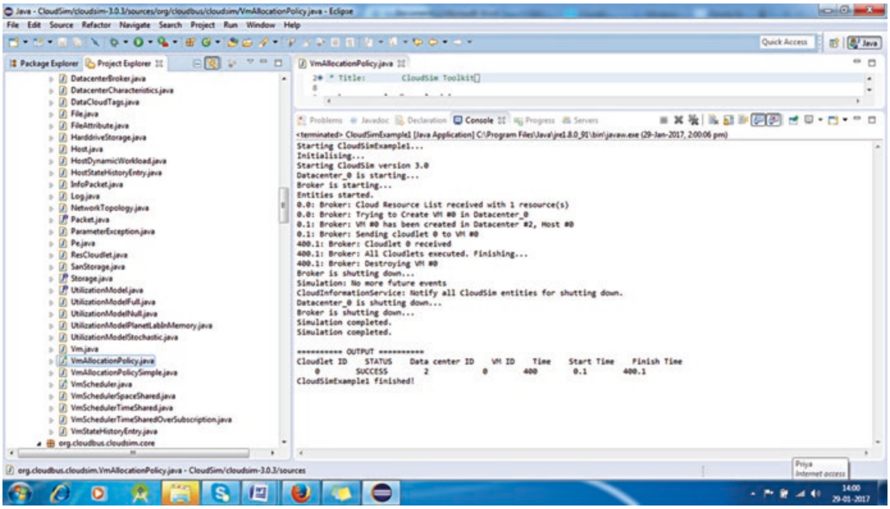
```
Starting CloudSimExample4...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Datacenter_1 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 2 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.0: Broker: Trying to Create VM #1 in Datacenter_0
[VmScheduler.vmCreate] Allocation of VM #1 to Host #0 failed by MIPS
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Creation of VM #1 failed in Datacenter #2
0.1: Broker: Trying to Create VM #1 in Datacenter_1
0.2: Broker: VM #1 has been created in Datacenter #3, Host #0
0.2: Broker: Sending cloudlet 0 to VM #0
0.2: Broker: Sending cloudlet 1 to VM #1
160.2: Broker: Cloudlet 0 received
```

```
160.2: Broker: Cloudlet 1 received
160.2: Broker: All Cloudlets executed. Finishing...
160.2: Broker: Destroying VM #0
160.2: Broker: Destroying VM #1
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Datacenter_1 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

========== OUTPUT ==========
Cloudlet ID  STATUS    Data center ID  VM ID  Time  Start Time  Finish Time
0            SUCCESS   2               0      160   0.2         160.2
1            SUCCESS   3               1      160   0.2         160.2
CloudSimExample4 finished!
BUILD SUCCESSFUL (total time: 11 seconds)
```

Example 2: VM allocation policy in cloud
Power-Aware VM Scheduling in CloudSim

```
Input:hostList,VMList  Output: allocati of VMs
VMList.sortDecreasingUtilization()
for each VM in VMList do
    minpower←MAX
    allocatedHost←null
    foreach host in hostList do
            if host has enough resources for VM
                    power←estimatePower(host,VM)
                            ifpower<manpower
                            allocatedHost←host
                            minpower← power
                    if(allocatedHost!=null) then
                            allocationadd(VM,allocatedHost)
return allocation
```

Screenshot for VM allocation policy in cloud



Java - CloudSim/cloudsim-3.0.3/sources/org/cloudbus/cloudsim/VmAllocationPolicy.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
Java
Package Explorer
Project Explorer
DatacenterBroker.java
DatacenterCharacteristics.java
DataCloudTags.java
File.java
FileAttribute.java
HarddriveStorage.java
Host.java
HostDynamicWorkload.java
HostStateHistoryEntry.java
InfoPacket.java
Log.java
NetworkTopology.java
Packet.java
ParameterException.java
Pe.java
ResCloudlet.java
SanStorage.java
Storage.java
UtilizationModel.java
UtilizationModelFull.java
UtilizationModelNull.java
UtilizationModelPlanetLabInMemory.java
UtilizationModelStochastic.java
Vm.java
VmAllocationPolicy.java
VmAllocationPolicySimple.java
VmScheduler.java
VmSchedulerSpaceShared.java
VmSchedulerTimeShared.java
VmSchedulerTimeSharedOverSubscription.java
VmStateHistoryEntry.java
org.cloudbus.cloudsim.core
VmAllocationPolicy.java
Problems
Javadoc
Declaration
Console
Progress
Servers
org.cloudbus.cloudsim.VmAllocationPolicy.java - CloudSim/cloudsim-3.0.3/sources
14:00
29-01-2017

```
2⊕ * Title:        CloudSim Toolkit
```

```
<terminated> CloudSimExample1 [Java Application] C:\Program Files\Java\jre1.8.0_91\bin\javaw.exe (29-Jan-2017, 2:00:06 pm)
Starting CloudSimExample1...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 1 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.1: Broker: VM #0 has been created in Datacenter #2, Host #0
0.1: Broker: Sending cloudlet 0 to VM #0
400.1: Broker: Cloudlet 0 received
400.1: Broker: All Cloudlets executed. Finishing...
400.1: Broker: Destroying VM #0
Broker is shutting down...
Simulation: No more future events
CloudInformationService: Notify all CloudSim entities for shutting down.
Datacenter_0 is shutting down...
Broker is shutting down...
Simulation completed.
Simulation completed.

========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID    Time    Start Time    Finish Time
    0        SUCCESS        2            0        400       0.1          400.1
CloudSimExample1 finished!
```

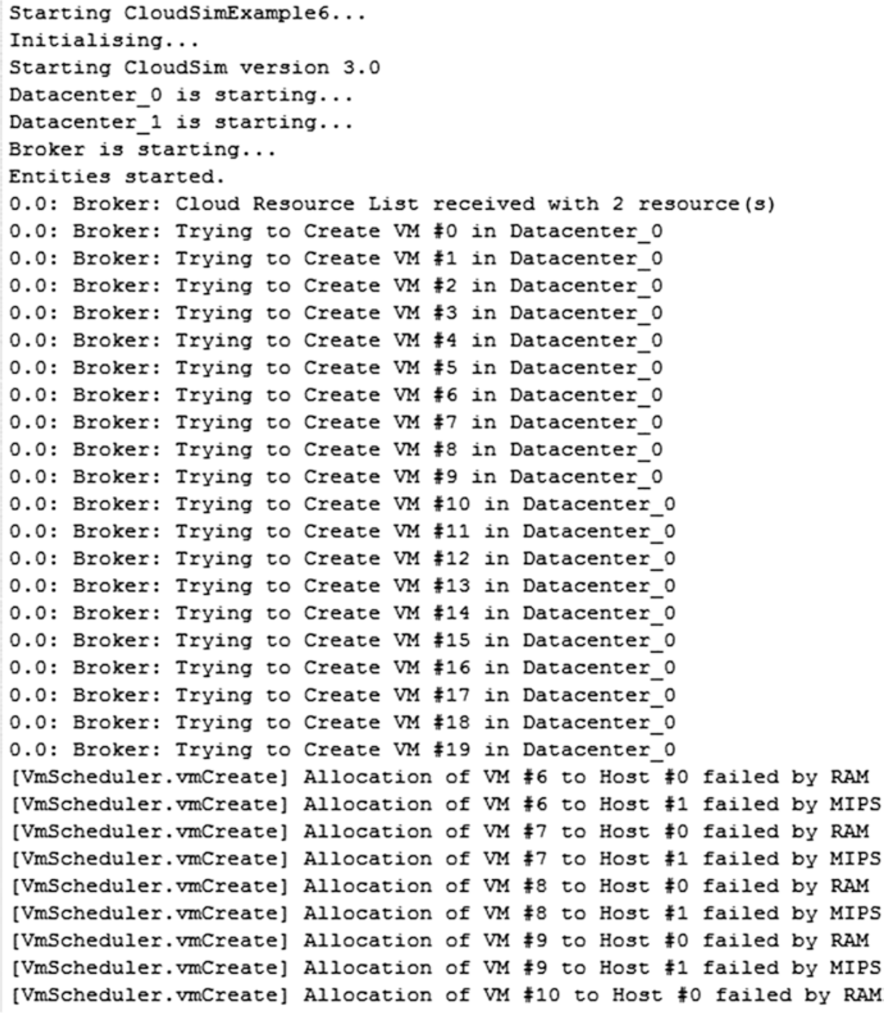
Example 3: Here 20 VMs are requested and 11 VMs are created successfully. Forty cloudlets are sent for execution. This output is for SpaceShared Policy. The total execution time of all cloudlets is 4.2.

Cloudlet Parameters Assigned

```
//cloudlet parameters
long length = 1000;
long fileSize = 300;
long outputSize = 300;
int pesNumber = 1;
```

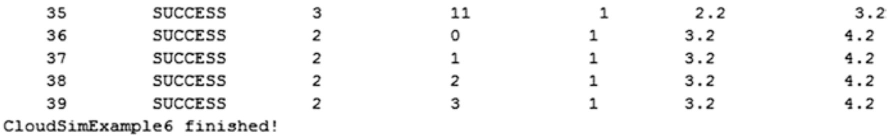
VM Parameters Assigned

```
//VM Parameters
long size = 10000; //image size (MB)
int ram = 512; //vm memory (MB)
int mips = 1000;
long bw = 1000;
int pesNumber = 1; //number of cpus
String vmm = "Xen"; //VMM name
```



```
Starting CloudSimExample6...
Initialising...
Starting CloudSim version 3.0
Datacenter_0 is starting...
Datacenter_1 is starting...
Broker is starting...
Entities started.
0.0: Broker: Cloud Resource List received with 2 resource(s)
0.0: Broker: Trying to Create VM #0 in Datacenter_0
0.0: Broker: Trying to Create VM #1 in Datacenter_0
0.0: Broker: Trying to Create VM #2 in Datacenter_0
0.0: Broker: Trying to Create VM #3 in Datacenter_0
0.0: Broker: Trying to Create VM #4 in Datacenter_0
0.0: Broker: Trying to Create VM #5 in Datacenter_0
0.0: Broker: Trying to Create VM #6 in Datacenter_0
0.0: Broker: Trying to Create VM #7 in Datacenter_0
0.0: Broker: Trying to Create VM #8 in Datacenter_0
0.0: Broker: Trying to Create VM #9 in Datacenter_0
0.0: Broker: Trying to Create VM #10 in Datacenter_0
0.0: Broker: Trying to Create VM #11 in Datacenter_0
0.0: Broker: Trying to Create VM #12 in Datacenter_0
0.0: Broker: Trying to Create VM #13 in Datacenter_0
0.0: Broker: Trying to Create VM #14 in Datacenter_0
0.0: Broker: Trying to Create VM #15 in Datacenter_0
0.0: Broker: Trying to Create VM #16 in Datacenter_0
0.0: Broker: Trying to Create VM #17 in Datacenter_0
0.0: Broker: Trying to Create VM #18 in Datacenter_0
0.0: Broker: Trying to Create VM #19 in Datacenter_0
[VmScheduler.vmCreate] Allocation of VM #6 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #6 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #7 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #7 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #8 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #8 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #9 to Host #0 failed by RAM
[VmScheduler.vmCreate] Allocation of VM #9 to Host #1 failed by MIPS
[VmScheduler.vmCreate] Allocation of VM #10 to Host #0 failed by RAM
```

The execution continues for all the cloudlets assigned with necessary VM allocation, and the simulation is successfully completed:



```
    35         SUCCESS        3            11            1        2.2          3.2
    36         SUCCESS        2            0            1        3.2          4.2
    37         SUCCESS        2            1            1        3.2          4.2
    38         SUCCESS        2            2            1        3.2          4.2
    39         SUCCESS        2            3            1        3.2          4.2
CloudSimExample6 finished!
```

5.5 WorkflowSim

Scientific workflows [10] are generally used for representing complex distributed scientific computations. They are having huge number of tasks and the execution of those tasks demands several complex modules and software. The performance validation of workflow optimization techniques in real infrastructures is difficult and time-consuming. Consequently, simulation-based approaches evolved as one among the most renowned approaches to validate the scientific workflows. Further, it relaxes the intricacy of the empirical setup and accumulates much effort in workflow execution by empowering the examination of their applications in a repeatable and controlled environment [10].

Current workflow simulators do not furnish a framework that considers the system overheads and failures. Further, they are not equipped with most used workflow optimization approaches such as task clustering [10]. WorkflowSim is an *open-source* workflow simulator, and it inherits CloudSim by providing a workflow-level support of simulation [11]. CloudSim supports only the execution of single workload, whereas WorkflowSim focuses on workflow scheduling as well as execution. CloudSim has a basic model of task execution which never considers the association among the tasks or clustering. Usually, it simply overlooks the unfortunate occurrence of failures and overheads [10].

WorkflowSim models workflows as a DAG model and provides an elaborate model for node failures, a model for delays occurring in the various levels of the workflow management system stack. It also includes implementations of several most popular dynamic and static workflow schedulers like HEFT and Min-Min and task clustering algorithms such as runtime-based algorithms, data-oriented algorithms, and fault-tolerant clustering algorithms [11].

5.6 Architecture of WorkflowSim

Figure 5.5 [10] depicts the architecture of the WorkflowSim. It clearly shows the various components contained in WorkflowSim, and the area surrounded by red lines is supported by CloudSim.

Workflow Mapper
Workflows are modeled as Directed Acyclic Graphs (DAGs), where nodes represent the tasks which are going to be executed over the computing resources and directed edges represent communication or control flow dependencies among the jobs [10]. The workflow mapper imports the XML-formatted DAG files and other metadata information. Once the mapping is completed, the workflow mapper produces a group of tasks and allocates them to an execution site. A task is a software or an action that a user would like to execute [11].

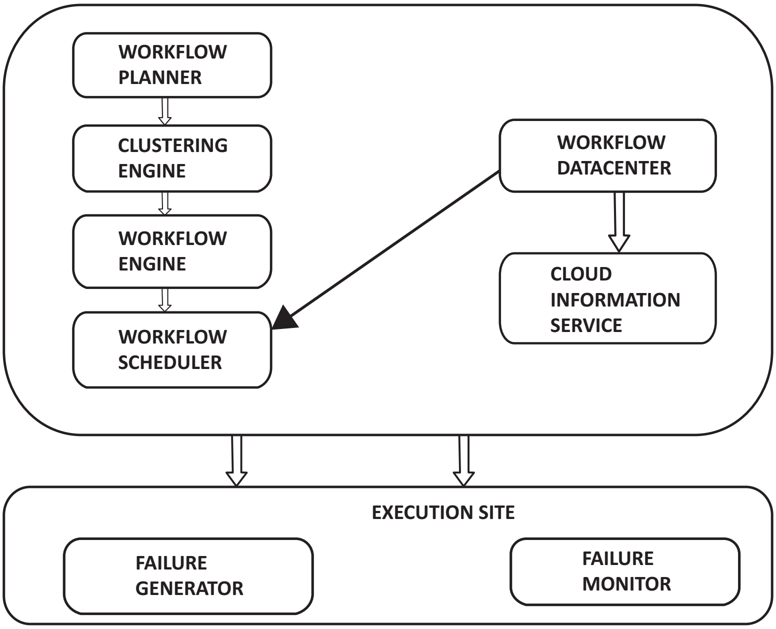


Fig. 5.5 WorkflowSim overview

Clustering Engine
A job is an infinitesimal unit which groups multiple tasks that are to be performed in sequence or in parallel. The clustering engine combines tasks into jobs so as to minimize the scheduling overheads [11]. There are two techniques in clustering such as horizontal clustering and vertical clustering. Horizontal clustering combines the tasks at the identical horizontal levels within the source workflow graph, and vertical clustering combines the tasks in an identical vertical pipeline levels [10]. Clustering engine [11] in WorkflowSim additionally performs task reclustering in a failure environment by means of transient failures. The failed jobs which are returned from the workflow scheduler are combined together to create a new job.

Workflow Engine
The workflow engine [11] selects the jobs to be executed based on the parent-child relationship. It controls jobs execution by considering their associations to ensure that a job is allowed to execute only when all of its parent jobs have completed successfully. The workflow engine will discharge free jobs to the workflow scheduler.

Workflow Scheduler
The workflow scheduler [11] is used to add the characteristics for the data center. It creates the resources like virtual machines in data center. It does the matching of jobs to appropriate resources based on the conditions chosen by users.

Interaction Between the Components
To combine and harmonize these components, an event-based approach [10] is adopted where each of them having a message queue. Figure 5.6 [10] showcases a

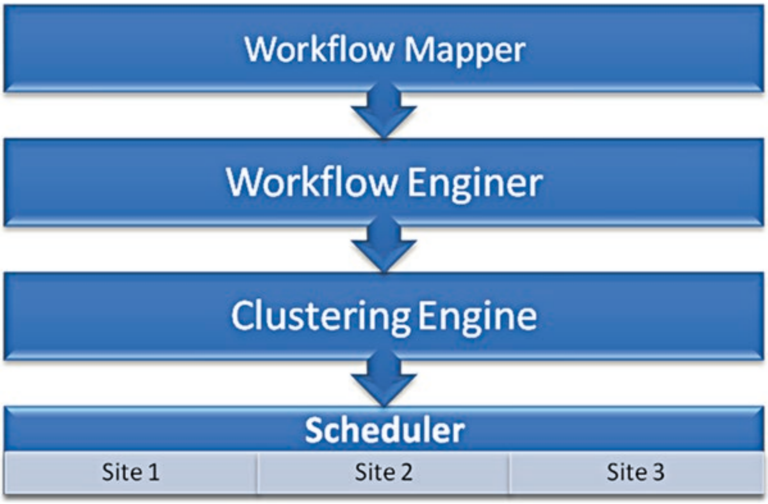


Fig. 5.6 Interaction between components

minimal configuration that contains two data centers and two nodes in each site. Here, each component manages its own message queue, and it periodically checks in a repeated manner, whether it needs to process any message.

For instance, the clustering engine checks every time whether it has received any new tasks from the workflow engine and whether any new jobs are to be released to the scheduler. When the message queue is empty for all the components, the simulation is completed.

Layered Failures and Job Retry

Failures may occur at a wide range of instances during the execution of workflows. These failures are broadly classified into two types: task failure and job failure. Task failure [10] occurs in a situation in which the transient failure affects the computation of a task and other tasks that are in the job list do not necessarily fail. Job failure [10] is a situation in which transient failure affects the clustered job and all the jobs in the task will fail eventually. There are two components added in response to the simulation of failures:

Failure Generator [10] component is used to insert task or job failures at each execution site. After each job execution, failure generator will randomly generate task/job failures based on the distribution and average failure rate that is specified by the user.

Failure Monitor [10] collects failure records such as resource id, job id, and task id. These records are then returned to the workflow management system to adjust the scheduling strategies dynamically.

Fault Tolerant Optimization There are several ways to improve the performance in a failure-prone environment. Two prominent ways are included in the WorkflowSim. The first method [10] emphasizes on the retry of the particular failed part of the job or the retry of the entire job. Workflow scheduler is used to take care of this functionality. Workflow scheduler will check the status of a job and takes actions based on the user-selected strategies. The second method [4] that can be used is reclustering. It is a technique in which the task clustering strategy is adjusted based on the detected failure rate. Workflow engine is used to take care of this functionality.

Existing Scheduling Algorithms

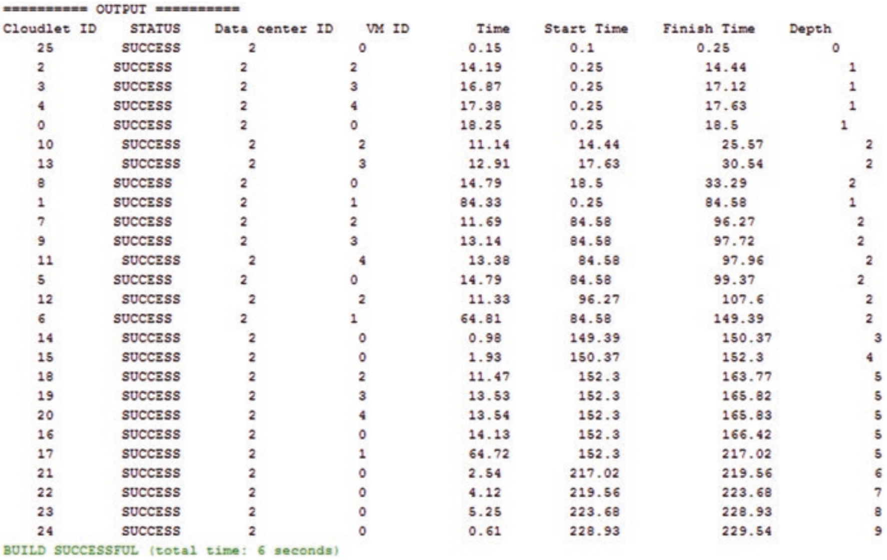
Example 1: First Come First Serve

First come first serve [10] is the basic version of scheduling algorithm used in workflow simulator. It assigns each job based on the arriving order of the jobs. It allocates the next available resources as the jobs are received. It does not take into consideration the jobs' expected completion time on that worker node. When there are multiple resources that are in the idle status, the resources are chosen randomly.

Pseudocode

```
for all cloudlets in the list
                check status of each VM from VmList
                assign vms to the cloudlets
```

Results



```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    25         SUCCESS        2              0          0.15       0.1           0.25          0
    2         SUCCESS        2              2          14.19      0.25          14.44           1
    3         SUCCESS        2              3          16.87      0.25          17.12           1
    4         SUCCESS        2              4          17.38      0.25          17.63           1
    0         SUCCESS        2              0          18.25      0.25          18.5           1
    10         SUCCESS        2              2          11.14      14.44          25.57            2
    13         SUCCESS        2              3          12.91      17.63          30.54            2
    8         SUCCESS        2              0          14.79      18.5          33.29           2
    1         SUCCESS        2              1          84.33      0.25          84.58           1
    7         SUCCESS        2              2          11.69      84.58          96.27           2
    9         SUCCESS        2              3          13.14      84.58          97.72           2
    11         SUCCESS        2              4          13.38      84.58          97.96            2
    5         SUCCESS        2              0          14.79      84.58          99.37           2
    12         SUCCESS        2              2          11.33      96.27          107.6            2
    6         SUCCESS        2              1          64.81      84.58          149.39           2
    14         SUCCESS        2              0          0.98       149.39         150.37            3
    15         SUCCESS        2              0          1.93       150.37         152.3            4
    18         SUCCESS        2              2          11.47      152.3          163.77            5
    19         SUCCESS        2              3          13.53      152.3          165.82            5
    20         SUCCESS        2              4          13.54      152.3          165.83            5
    16         SUCCESS        2              0          14.13      152.3          166.42            5
    17         SUCCESS        2              1          64.72      152.3          217.02            5
    21         SUCCESS        2              0          2.54       217.02         219.56            6
    22         SUCCESS        2              0          4.12       219.56         223.68            7
    23         SUCCESS        2              0          5.25       223.68         228.93            8
    24         SUCCESS        2              0          0.61       228.93         229.54            9
BUILD SUCCESSFUL (total time: 6 seconds)
```

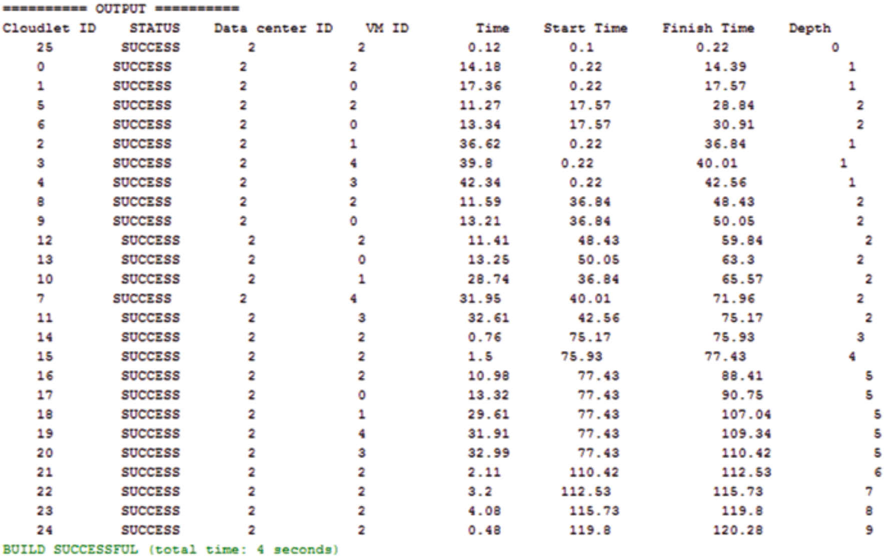
Example 2: Minimum Completion Time

Minimum completion time [10] assigns each job to the available resource based solely on the best expected completion time of the selected job.

Pseudocode

```
for each cloudlet in the list
   for each vm in VmList
      check for VM_STATUS_IDLE
      assign that vm as firstIdleVm
   for each vm in VmList
        check for VM_STATUS_IDLE
      compare each vm with the firstIdleVm for completion time
   change the vm status to VM_STATUS_BUSY
   assign vms to the cloudlets
```

Results



```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID       Time    Start Time    Finish Time    Depth
    25          SUCCESS        2            2           0.12      0.1           0.22           0
    0          SUCCESS        2            2          14.18      0.22          14.39            1
    1          SUCCESS        2            0          17.36      0.22          17.57            1
    5          SUCCESS        2            2          11.27      17.57          28.84            2
    6          SUCCESS        2            0          13.34      17.57          30.91            2
    2          SUCCESS        2            1          36.62      0.22          36.84            1
    3          SUCCESS        2            4          39.8      0.22          40.01            1
    4          SUCCESS        2            3          42.34      0.22          42.56            1
    8          SUCCESS        2            2          11.59      36.84          48.43            2
    9          SUCCESS        2            0          13.21      36.84          50.05            2
    12          SUCCESS        2            2           11.41      48.43          59.84           2
    13          SUCCESS        2            0           13.25      50.05          63.3            2
    10          SUCCESS        2            1           28.74      36.84          65.57           2
    7          SUCCESS        2            4          31.95      40.01          71.96            2
    11          SUCCESS        2            3           32.61      42.56          75.17           2
    14          SUCCESS        2            2           0.76      75.17          75.93            3
    15          SUCCESS        2            2           1.5      75.93          77.43            4
    16          SUCCESS        2            2           10.98      77.43          88.41           5
    17          SUCCESS        2            0           13.32      77.43          90.75           5
    18          SUCCESS        2            1           29.61      77.43          107.04           5
    19          SUCCESS        2            4           31.91      77.43          109.34           5
    20          SUCCESS        2            3           32.99      77.43          110.42           5
    21          SUCCESS        2            2           2.11      110.42          112.53           6
    22          SUCCESS        2            2           3.2      112.53          115.73           7
    23          SUCCESS        2            2           4.08      115.73          119.8            8
    24          SUCCESS        2            2           0.48      119.8          120.28           9
BUILD SUCCESSFUL (total time: 4 seconds)
```

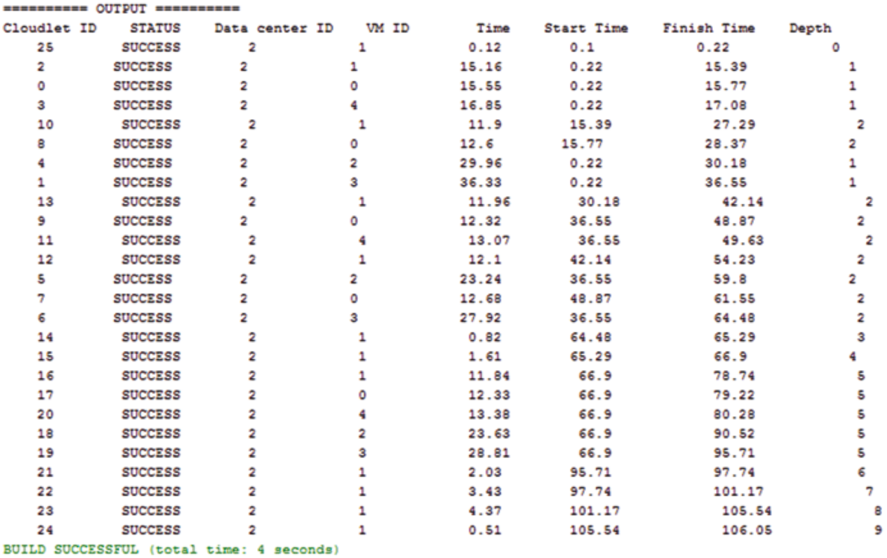
Example 3: Min-Min

The Min-Min [10] heuristic begins by sorting the free jobs based on the order of completion time. From the sorted order, the job with the minimum completion time is selected and allocated the corresponding resource. The next job is submitted to the queue and the process repeats until all the jobs in the list are scheduled. The main aim of Min-Min algorithm is to create a local optimal path such that the overall runtime is reduced.

Pseudocode

Results

```
for each cloudlet in the list
     select one cloudlet as minCloudlet
for each cloudlet
     Check length if lesser than length of minCloudlet
     Assign cloudlet as minCloudlet
     for each vm in VmList
            check for VM_STATUS_IDLE
              assign that vm as firstIdleVm
     for each vm in VmList
              check for VM_STATUS_IDLE
            compare each vm with the firstIdleVm for minimum
completion time
     change the vm status to VM_STATUS_BUSY
     assign firstIdleVm to the minCloudlet
```



```
========== OUTPUT ==========
Cloudlet ID    STATUS    Data center ID    VM ID        Time    Start Time    Finish Time    Depth
    25         SUCCESS        2              1          0.12       0.1           0.22            0
    2         SUCCESS        2              1          15.16      0.22          15.39            1
    0         SUCCESS        2              0          15.55      0.22          15.77            1
    3         SUCCESS        2              4          16.85      0.22          17.08            1
    10         SUCCESS        2              1          11.9      15.39          27.29            2
    8         SUCCESS        2              0          12.6      15.77          28.37            2
    4         SUCCESS        2              2          29.96      0.22          30.18            1
    1         SUCCESS        2              3          36.33      0.22          36.55            1
    13         SUCCESS        2              1          11.96      30.18          42.14            2
    9         SUCCESS        2              0          12.32      36.55          48.87            2
    11         SUCCESS        2              4          13.07      36.55          49.63            2
    12         SUCCESS        2              1          12.1      42.14          54.23            2
    5         SUCCESS        2              2          23.24      36.55          59.8            2
    7         SUCCESS        2              0          12.68      48.87          61.55            2
    6         SUCCESS        2              3          27.92      36.55          64.48            2
    14         SUCCESS        2              1          0.82      64.48          65.29            3
    15         SUCCESS        2              1          1.61      65.29          66.9            4
    16         SUCCESS        2              1          11.84      66.9          78.74            5
    17         SUCCESS        2              0          12.33      66.9          79.22            5
    20         SUCCESS        2              4          13.38      66.9          80.28            5
    18         SUCCESS        2              2          23.63      66.9          90.52            5
    19         SUCCESS        2              3          28.81      66.9          95.71            5
    21         SUCCESS        2              1          2.03      95.71          97.74            6
    22         SUCCESS        2              1          3.43      97.74          101.17            7
    23         SUCCESS        2              1          4.37      101.17          105.54            8
    24         SUCCESS        2              1          0.51      105.54          106.05            9
BUILD SUCCESSFUL (total time: 4 seconds)
```

5.7 Conclusion

CloudSim and WorkflowSim are increasingly popular to simulate the cloud computing environment due to their guarantee for dynamic repeatable evaluation of provisioning policies for several applications. Simulation frameworks achieve faster validation of scheduling and resource allocation mechanisms in cloud data centers. Further, diverse network topologies with several parallel applications can be modeled and simulated in NetworkCloudSim, which was recently developed [2, 3]. CloudSim is very useful tool for modeling and simulation of data centers with necessary resource allocation based on various scheduling algorithms. Furthermore, the WorkflowSim is used to execute DAG-like job that has an interdependency between them. Additionally, WorkflowSim can also be used to validate several scheduling and planning algorithms.

References

1. http://www.cloudbus.org/cloudsim/
2. Piraghaj SF, Dastjerdi AV, Calheiros RN, Buyya R (2016) Container CloudSim: a framework and algorithm for energy efficient container consolidation in cloud data centers, an environment for modeling and simulation of containers in cloud data centers, Software Pract Exp, John Wiley & Sons, Ltd, USA, 2016. (Accepted for publication)
3. Garg SK, Buyya R (2011) NetworkCloudSim: modelling parallel applications in cloud simulations. In: Proceedings of the 4th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2011, IEEE CS Press, USA), Melbourne, Australia, December 5–7, 2011
4. Calheiros RN, Ranjan R, Beloglazov A, De Rose CAF, Buyya R (2011) CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software Pract Exp 41(1):23–50. ISSN: 0038-0644, Wiley Press, New York, USA, January, 2011
5. Zheng J, Sun Y, Zohu W (2009) Cloud computing based internet data center. Springer Cloud Com pp 700–704
6. Sarishma KM (2015) Cloud storage: focusing on back end storage architecture. IOSR J Comput Eng 17(Jan–Feb):2278–8727
7. Sabahi F (2012) Member, IEEE – secure virtualization for cloud environment using hypervisor-based technology. Int J Mach Learn Comput 2(1)
8. Ferreira L, Putnik G, Cunha M (2013) Science direct – cloudlet architecture for dashboard in cloud and ubiquitous manufacturing vol 12
9. Raj G, Nischal A (2012) Efficient resource allocation in resource provisioning policies over resource cloud communication paradigm. Int J Cloud Comput Serv Architect 2(3):79–83
10. Chen W, Deelman E (2012) WorkflowSim: a toolkit for simulating scientific workflows in distributed environments. In: Proceedings of the 8th International Conference on E-Science (E-Science)
11. http://www.workflowsim.org/index.html

Chapter 6
Execution of Workflow Scheduling in Cloud Middleware

Abstract Many scientific applications are often modeled as workflows. The data and computational resource requirements are high for such workflow applications. Cloud provides a better solution to this problem by offering the promising environment for the execution of these workflow. As it involves tremendous data computations and resources, there is a need to automate the entire process. Workflow management system serves this purpose by orchestrating workflow task and executing it on distributed resources. Pegasus is a well-known workflow management system that has been widely used in large-scale e-applications. This chapter provides an overview about the Pegasus Workflow Management System, describes the environmental setup with OpenStack and creation and execution of workflows in Pegasus, and discusses about the workflow scheduling in cloud with its issues.

6.1 Introduction

Applications such as multi-tier web service workflows [1], scientific workflows [2], and big data processing like MapReduce consist of numerous dependent tasks that require a lot of computational power. This massive computational power requirement makes these applications difficult to process. An effective means to define such multifarious applications is achieved through workflow scheduling algorithms and frameworks. The problem of workflow scheduling in the distributed environments has been addressed widely in the literature. Harshad Prajapathi [3] discussed scheduling in grid computing environment and presented a concise perceptive about the grid computing system. Fuhui Wu et al. presented a workflow scheduling algorithm in cloud environment and also presented a comparative review on workflow scheduling algorithm [4]. Grid computing-based workflow scheduling taxonomy was proposed by Jia yu et al. [5]. Mohammad Masdari (Masdari et al.) [6] presented a comprehensive survey and analysis of workflow scheduling in cloud computing.

Workflow applications are represented as Directed Acyclic Graphs (DAGs), and the scheduling of workflow is a NP-complete problem [7]. These applications require higher computing environment for the execution of their workflows. Traditional high-performance computing (HPC) support with cloud computing will be the effective solution for the execution of these intensive computational power demanding workflows. Many definitions have been proposed for cloud computing

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_6

[8, 9] According to NIST [10] Peter M. Mell and Timothy Grance, cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Cloud computing is a popular concept that has emerged from various heterogeneous distributed computing such as utility computing, grid computing, and autonomic computing. Cloud computing is viewed as a solution to effectively run workflows and has rapidly gained the interest of researchers in scientific community. Cloud provides resources as services over a network in an on-demand fashion, and the user needs to identify the resource type to lease, lease time, and cost for running the application. There are several open-source clouds available for creating private and public clouds. OpenStack cloud environment-based experimental analysis is carried out for various scenarios in this chapter.

The prospect of running workflow applications through cloud is made attractive by its very many benefits. The essential benefits include:

- Virtualization

Cloud gives the illusion of unlimited resources, and this allows the user to acquire sufficient resources at any time.

- Elasticity

Cloud allows its user to scale up and scale down the resources by acquiring and releasing the resources as and when required.

The major contributions include:

- A description of an approach that sets up a computational environment in OpenStack cloud to support the execution of scientific workflow applications
- Creation of workflows with Pegasus-keg tool and their execution in OpenStack cloud
- A comparison of the performance of workflow applications using different instances of OpenStack
- Addressing of some of the issues in scheduling workflows in cloud

Many works have been carried out for workflows in cloud, and Google scholar reports 17,700 entries for the keyword workflows and cloud from 2010 to 2016. This shows that workflow in cloud is a significant area of research.

Many scientific applications are often modeled as workflows. The data and computational resource requirements are high for such workflow applications. Cloud provides a better solution to this problem by offering a promising environment for the execution of these workflows. Modeling as workflows involves tremendous data computations and resources, thereby creating the necessity to automate the entire process. Workflow management system serves this purpose by orchestrating the workflow tasks and by executing it on distributed resources. Pegasus is a well-known workflow management system that has been widely used in large-scale e-applications. This chapter provides an overview of the Pegasus Workflow

Management System and gives the description of the environmental setup with OpenStack.

This chapter is organized as follows: Section 6.2 describes the workflow management system with the overview of Pegasus and its subsystems. Section 6.3 discusses about the execution environmental setup for the experiment with the creation and execution of workflow in Pegasus. Section 6.4 describes scheduling algorithm and addresses the issues of scheduling workflow in cloud. Finally Sect. 6.5 concludes the chapter.

6.2 Workflow Management System

A scientific workflow describes the necessary computational steps and their data dependency for accomplishing a scientific objective. The tasks in the workflows are organized and orchestrated according to dataflow and dependencies. An appropriate administration is needed for the efficient organization of the workflow applications. Workflow management system (WMS) is used for proper management of workflow execution on computing resources. Pegasus is such an open-source workflow management system that consists of different integrated technologies that provides for execution of workflow-based applications in heterogeneous environments and manages the workflow running on potentially distributed data and compute resources. Architecture of the Pegasus WMS is shown in the Fig. 6.1.

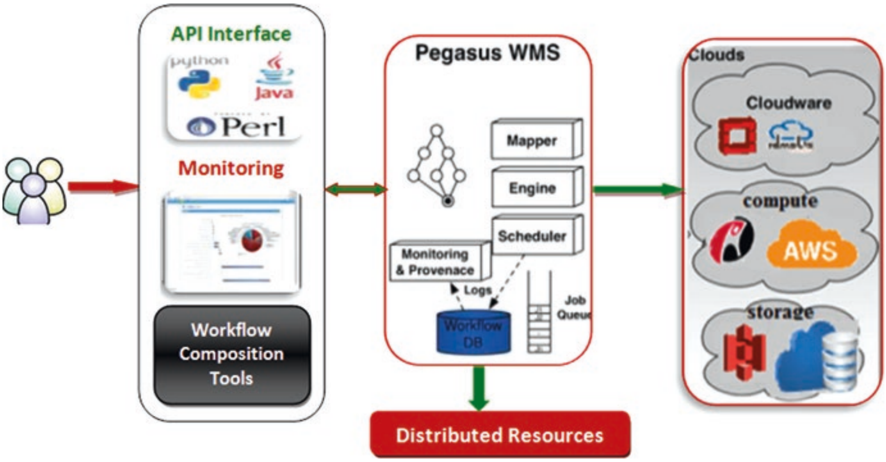


Fig. 6.1 Pegasus workflow Management System architecture [11]

Table 6.1 Major subsystems of the Pegasus workflow management system [12]

Mapper	Generates an executable workflow from the abstract workflows which was provided by the user. Mapper finds the appropriate resources and other related components which are required for the workflow execution. It also restructures workflows for optimized performance
Local execution engine	Submits the jobs and manages them by tracking the job state and determines when to run a job. Then it submits the jobs to the local scheduling queue
Job scheduler	It manages the jobs on both local and remote resources
Remote execution engine	It manages the execution of jobs on the remote computing nodes

In Pegasus, workflows are represented as DAGs (Directed Acyclic Graphs) by the users. DAG consists of nodes and edges, where the computational tasks are represented as nodes and the control flows and dataflows are represented as edges. Workflows are submitted as abstract workflows, that is, they do not contain any information about the resources or location of data. The workflows are submitted to Pegasus which resides on a machine called a submit host.

Pegasus links the scientific domain and the execution environment by mapping the abstract workflow description to an executable workflow description for computation. It consists of five major subsystems for performing various actions. Table 6.1 lists these major subsystems and their descriptions.

Monitoring component – it monitors the workflow and keeps track of all the information and logs. Workflow database collects those information and provides the provenance and performance information. Additionally it notifies the user about the workflow status.

6.3 Experimental Setup

OpenStack cloud is used for the experiment. Presently no other work has been carried out in the OpenStack with Pegasus WMS. Till date all the experiments have been done with AmazonEC2 instances. There are numerous ways to configure an execution environment for workflow application in cloud. The environment can be deployed entirely in the cloud, or parts of it can reside outside the cloud. The former approach has been chosen for this experiment and the private cloud setup has been accomplished with OpenStack. Instances are created with different flavors such as medium, large, xlarge, and dxlarge with CentOS 6.5 as a base operating system in addition with Pegasus and HTCondor. The information regarding the resource usage of this experiment is detailed in Table 6.2. To reduce the setup time, preconfigured images are used to create instances in cloud. The Pegasus Workflow Management System environmental setup in the OpenStack cloud is shown in Fig. 6.2.

Workflow applications are loosely coupled parallel applications which comprise of a set of tasks that are linked with dataflow and control-flow dependencies.

Table 6.2 Resource type used

Instance name	Flavor	Architecture	Core	Memory RAM (GB)
Pegasus-med	m1.medium	x_86	2	4
Pegasus-l	m1.large	x_86	4	8
Pegasus-x	m1.xlarge	x_86	8	16
Pegasus-dx	m1.dxlarge	x_86	16	32
Localhost.localdomain	Desktop machine	x_86	4	2

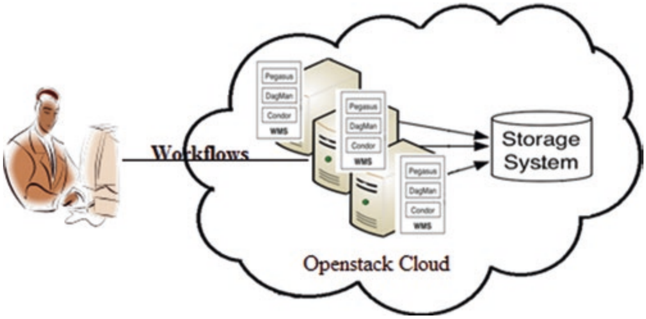


Fig. 6.2 Pegasus workflow management system execution environment in OpenStack

Workflow tasks use file system for communication between them. Each task produces one or more outputs which becomes the input for other tasks. Workflows are executed with Pegasus Workflow Management System with DAGMan and Condor. Pegasus transforms the abstract workflow into concrete plans which are executed using DAGMan to manage task dependencies and Condor to manage task execution.

A study of Pegasus environment with its control flow has been made by executing the example workflows that are shipped with Pegasus in different instances. The results for the execution of examples in different OpenStack instances are given in Fig. 6.3.

Desktop machine refers to the standalone machine, and others are the instances in OpenStack cloud. From the results, we infer that the performances of the OpenStack instances are better than the ones on the standalone machine. Hence, cloud instances are preferred for execution of the workflows than standalone.

6.4 General Steps for Submitting Workflow in Pegasus

1. Specify the computation in terms of DAX (Directed Acyclic Graph in XML). Write a simple DAX generator.
2. Set up catalogs.

   TC – Transformation Catalog

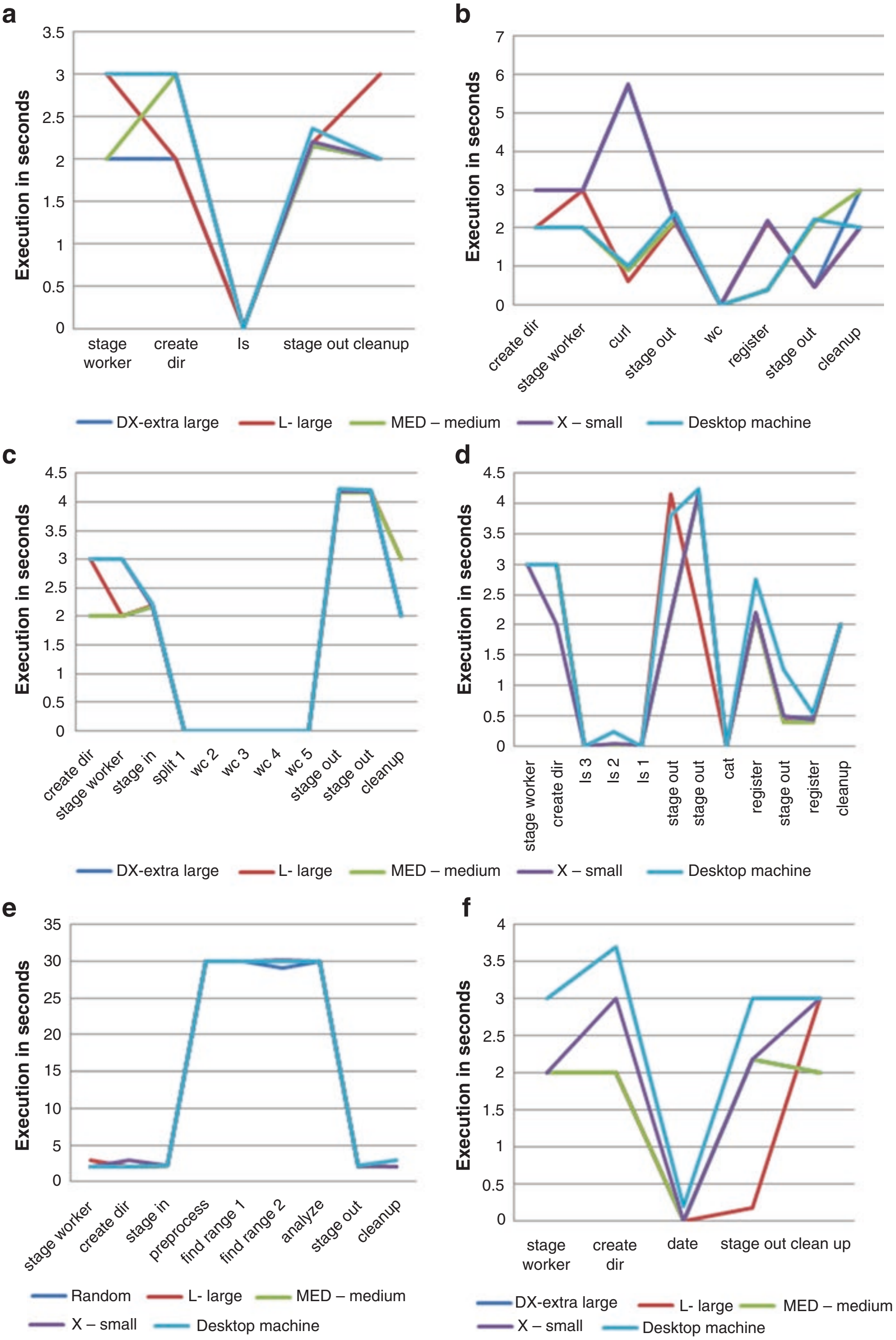


Fig. 6.3 Workflow execution in different OpenStack instances. (**a**) Process workflow execution. (**b**) Pipeline workflow execution. (**c**) Split workflow execution. (**d**) Merge workflow execution. (**e**) Diamond workflow execution. (**f**) Date workflow execution

Transformation Catalog maps logical transformations to physical executables on the system

RC – Replica Catalog

Replica Catalog maps logical file ids/names (LFNs) to physical file ids/names (PFNs).

Site catalog

Site catalog describes the computing resource where the workflows are to be executed. It is described in XML.

3. Planning and submission of workflows

   Pegasus-plan generates executable workflow maps this onto the target resources and submits it for execution.

4. Monitoring and analysis of workflows

   Pegasus-status is used to monitor the workflow execution.
   Pegasus-analyzer monitors and analyzes your workflow.

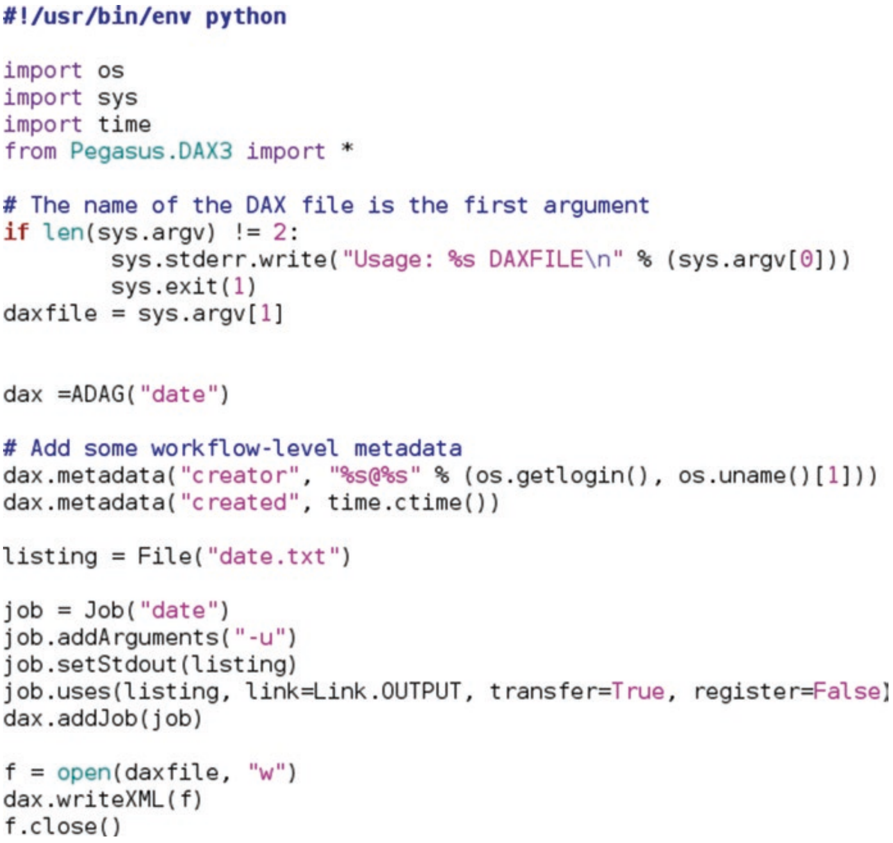
5. Mine workflows

   Pegasus-statistics is used to generate the statistical report about the workflow.
   Pegasus-plots is used to generate the graphs and charts to visualize the workflow.

To provide a simpler and better perceptive to the reader, the creation and submission of a simple workflow called "date" in OpenStack cloud using Pegasus is described.

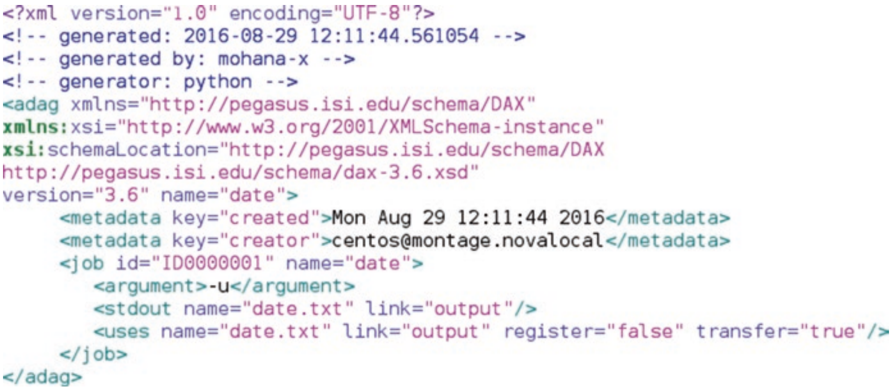
A date workflow is created with a single task that runs date command and puts the output in a text file. Date command uses the argument –u to display the UTC time. The abstract workflow (DAX) provides the description about the workflow without the resource dependency. It consists of the task for computation, computational order, required inputs, expected outputs, and arguments which are needed for the task invocation. Execution order will be represented as edges in DAG. The python DAX generator for date workflow is shown in Fig. 6.4. The generated DAX is shown in the Fig. 6.5. In this example, ID0000001 refers to the date executable which is identified by the tuple. It generates the output and places it in date.txt.

Once mapper decides where to retrieve data from and where to execute the job, it starts generating the executable workflow, that is, the abstract nodes are transformed into the nodes that contain executable jobs. Data management tasks are added, and these represent the data staging of the required input data, staging out of the output products back to the storage systems, and cataloging for future data discovery. In order to transform the abstract workflow into executable workflow, Pegasus requires information about the location of data, computational resources and workflow executables, storage systems, and schedulers. Pegasus queries the catalogs (site, replica, and transformation) to obtain this information, and then it converts the abstract workflow into executable workflow by including computational resources and data management steps. Figure 6.6 shows the transformation of abstract workflow to executable workflow.



```
#!/usr/bin/env python

import os
import sys
import time
from Pegasus.DAX3 import *

# The name of the DAX file is the first argument
if len(sys.argv) != 2:
        sys.stderr.write("Usage: %s DAXFILE\n" % (sys.argv[0]))
        sys.exit(1)
daxfile = sys.argv[1]


dax =ADAG("date")

# Add some workflow-level metadata
dax.metadata("creator", "%s@%s" % (os.getlogin(), os.uname()[1]))
dax.metadata("created", time.ctime())

listing = File("date.txt")

job = Job("date")
job.addArguments("-u")
job.setStdout(listing)
job.uses(listing, link=Link.OUTPUT, transfer=True, register=False)
dax.addJob(job)

f = open(daxfile, "w")
dax.writeXML(f)
f.close()
```

Fig. 6.4 Python DAX generator for date workflow



```
<?xml version="1.0" encoding="UTF-8"?>
<!-- generated: 2016-08-29 12:11:44.561054 -->
<!-- generated by: mohana-x -->
<!-- generator: python -->
<adag xmlns="http://pegasus.isi.edu/schema/DAX"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://pegasus.isi.edu/schema/DAX
http://pegasus.isi.edu/schema/dax-3.6.xsd"
version="3.6" name="date">
    <metadata key="created">Mon Aug 29 12:11:44 2016</metadata>
    <metadata key="creator">centos@montage.novalocal</metadata>
    <job id="ID0000001" name="date">
       <argument>-u</argument>
       <stdout name="date.txt" link="output"/>
       <uses name="date.txt" link="output" register="false" transfer="true"/>
    </job>
</adag>
```

Fig. 6.5 Date DAX

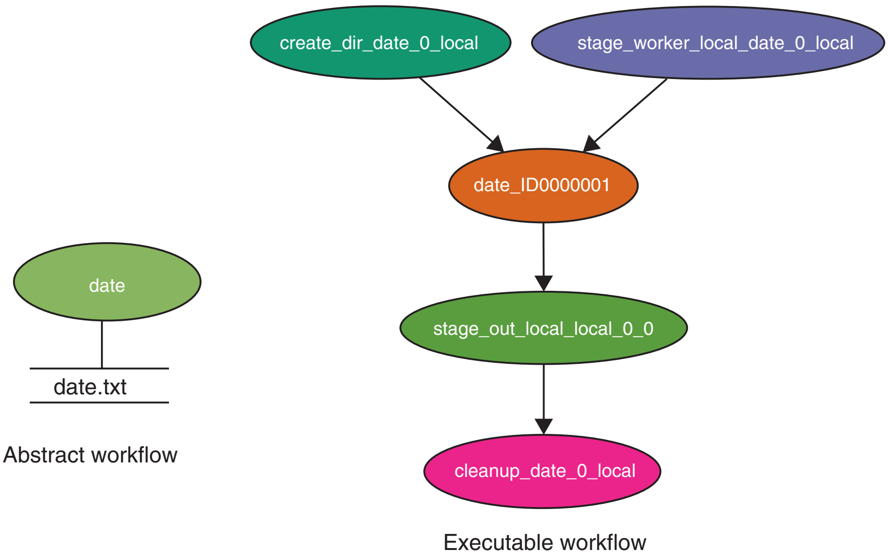
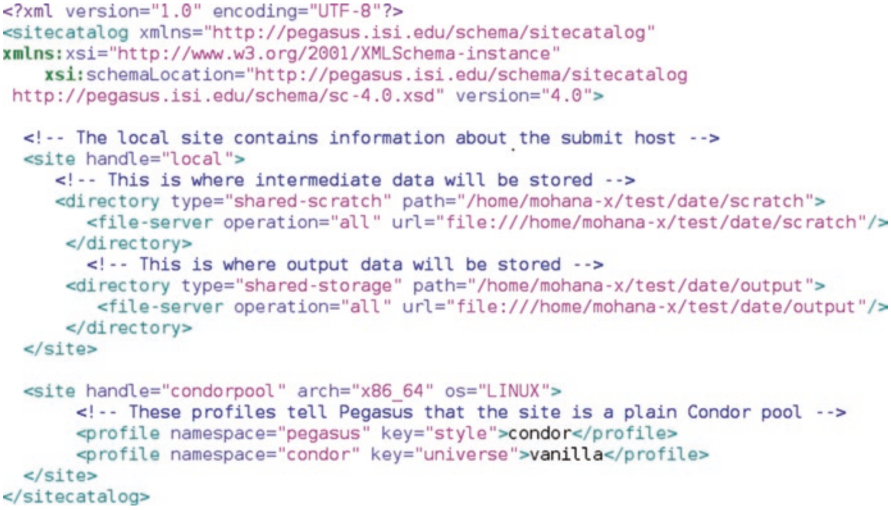


Fig. 6.6 Translation of a simple date DAX to an executable workflow



```
<?xml version="1.0" encoding="UTF-8"?>
<sitecatalog xmlns="http://pegasus.isi.edu/schema/sitecatalog"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://pegasus.isi.edu/schema/sitecatalog
 http://pegasus.isi.edu/schema/sc-4.0.xsd" version="4.0">

  <!-- The local site contains information about the submit host -->
  <site handle="local">
     <!-- This is where intermediate data will be stored -->
     <directory type="shared-scratch" path="/home/mohana-x/test/date/scratch">
        <file-server operation="all" url="file:///home/mohana-x/test/date/scratch"/>
      </directory>
        <!-- This is where output data will be stored -->
      <directory type="shared-storage" path="/home/mohana-x/test/date/output">
         <file-server operation="all" url="file:///home/mohana-x/test/date/output"/>
      </directory>
  </site>

  <site handle="condorpool" arch="x86_64" os="LINUX">
       <!-- These profiles tell Pegasus that the site is a plain Condor pool -->
       <profile namespace="pegasus" key="style">condor</profile>
       <profile namespace="condor" key="universe">vanilla</profile>
  </site>
</sitecatalog>
```

Fig. 6.7 Sites.xml

The computational resources are identified as site in site catalog. Here the site is local Condor [13] where the submitted workflow will run. Figure 6.7 shows the site catalog for the date workflow. Executables are identified from the transformation catalog. After mapping process gets completed, workflow is generated with a specified target workflow engine. Then it will be submitted to the engine and to its scheduler on the submit host. HTCondor DAGMan (Thain et al., 2005) is the default

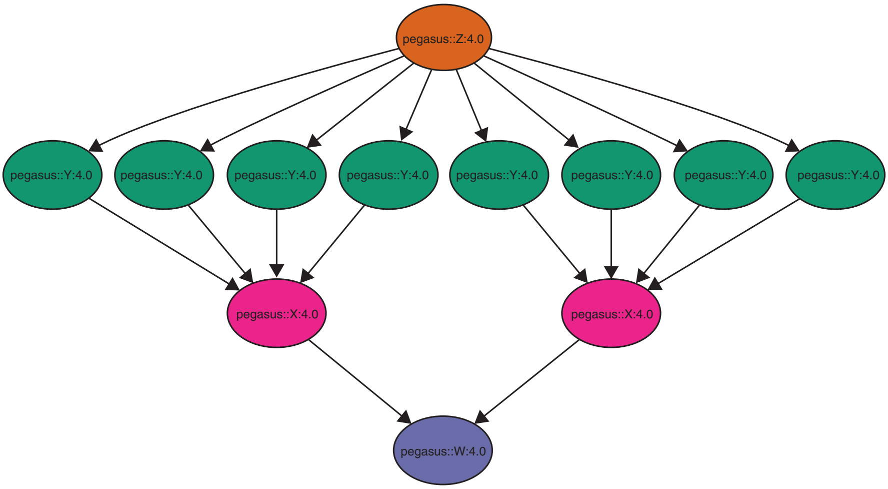


Fig. 6.8 Abstract workflow for the test generated by Pegasus using Pegasus-keg

workflow engine and HTCondor schedd is the default scheduler. The workflow is planned with Pegasus-plan for submission of the same to the execution host.

The test workflows for the execution can be generated with the help of Pegasus-keg (canonical executable) that is installed along with the installation of Pegasus WMS. It is used to create the desired workflow task that runs for a specified time and generates a specified size output. Pegasus-keg has several options to generate workflows with desired features such as:

- i: To read all input files into memory buffer
- o: To write the content to output files
- T: To generate CPU load for specified time
- a: To specify the name of the application

Test workflow with 12 nodes is created for the execution in Pegasus WMS with the help of Pegasus-keg. Abstract workflows are prepared in dax format with a use of java API generator, which is a primary input to Pegasus. The created test workflow structure is shown in Fig. 6.8, and this is an abstract workflow. Figure 6.9 shows the concrete workflow which can be executed over a number of resources. Here the workflow is planned to be executed in a cloud environment, and the site handle is condorpool. Output schedule for the test workflow is shown in Fig. 6.10.

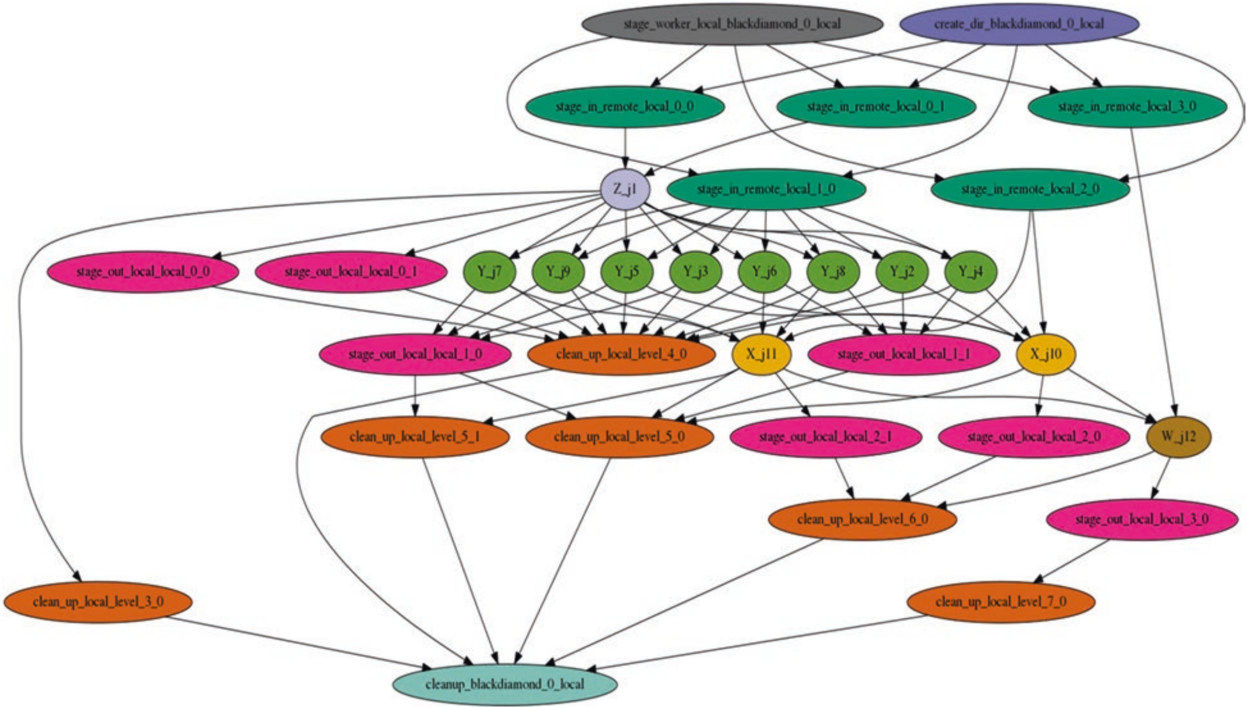

stage_worker_local_blackdiamond_0_local
create_dir_blackdiamond_0_local
stage_in_remote_local_0_0
stage_in_remote_local_0_1
stage_in_remote_local_3_0
Z_j1
stage_in_remote_local_1_0
stage_in_remote_local_2_0
stage_out_local_local_0_0
stage_out_local_local_0_1
Y_j7
Y_j9
Y_j5
Y_j3
Y_j6
Y_j8
Y_j2
Y_j4
stage_out_local_local_1_0
clean_up_local_level_4_0
X_j11
stage_out_local_local_1_1
X_j10
clean_up_local_level_5_1
clean_up_local_level_5_0
stage_out_local_local_2_1
stage_out_local_local_2_0
W_j12
clean_up_local_level_6_0
stage_out_local_local_3_0
clean_up_local_level_3_0
clean_up_local_level_7_0
cleanup_blackdiamond_0_local

Fig. 6.9 Translation of DAX to an executable DAG

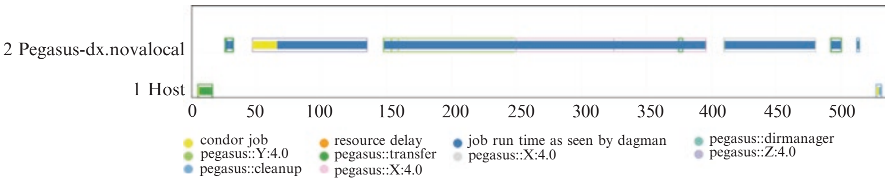


Fig. 6.10 The output schedule for the test workflow of 12 tasks scheduled provided by Pegasus WMS

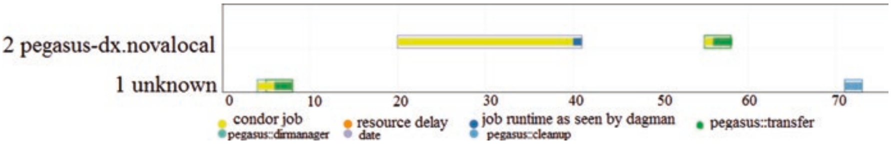


Fig. 6.11 Output schedule for date workflow scheduled using random site selector

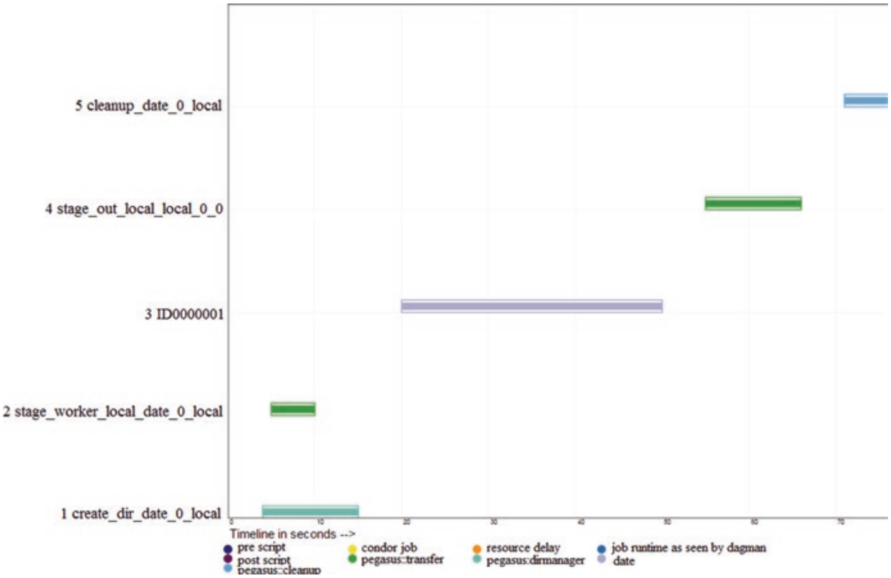


Fig. 6.12 Workflow execution Gantt chart for date workflow

6.4.1 *Pegasus Monitoring and Measuring Service*

Pegasus service is used to monitor the workflows running in Pegasus Workflow Management System which can be started by using Pegasus-service command (Figs. 6.11 and 6.12). Pegasus dashboard and status for the example workflows executed in Pegasus are given in the Figs. 6.13 and 6.14. Pegasus-plots is a tool used

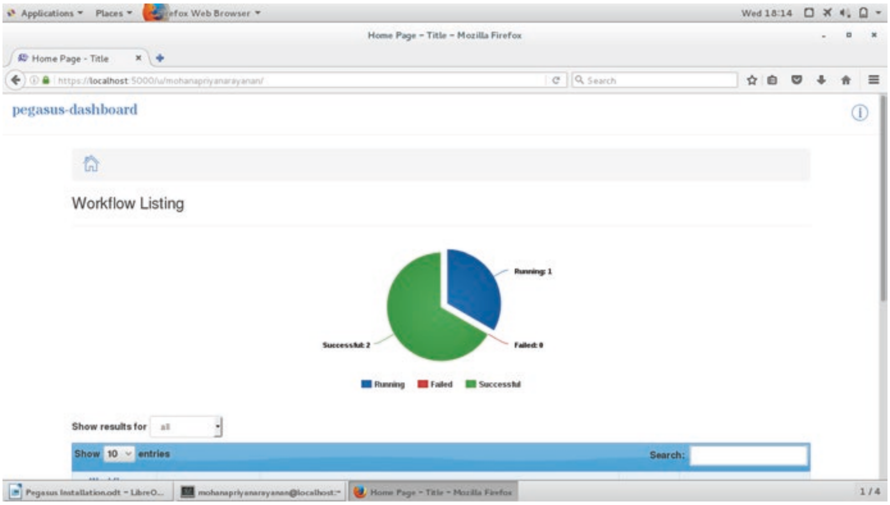
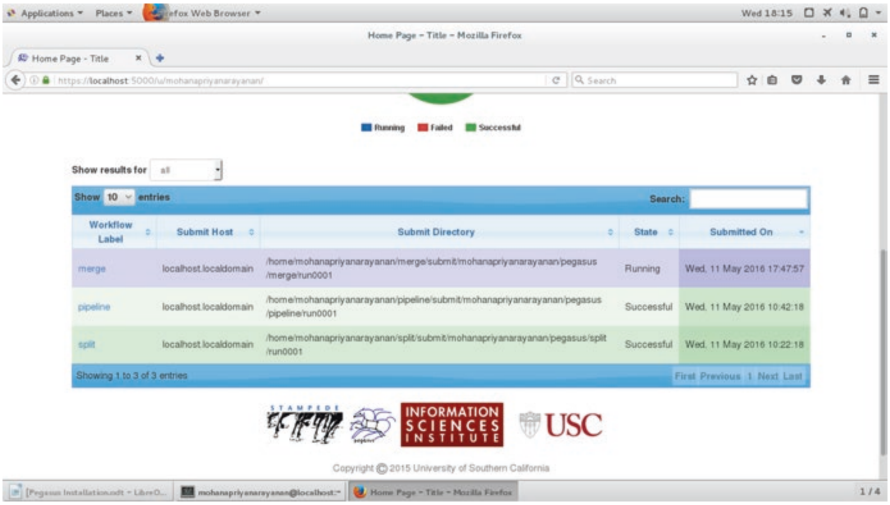


Fig. 6.13 Pegasus Dashboard – workflow execution in Pegasus WMS


Applications
Places
Firefox Web Browser
Wed 18:15
Home Page - Title - Mozilla Firefox
Home Page - Title
https://localhost:5000/u/mohanapriyanarayanan/
Search
Running
Failed
Successful
Show results for
all
Show 10 entries
Search:
First Previous 1 Next Last
STAMPEDE
INFORMATION SCIENCES INSTITUTE
USC
Pegasus Installation.odt - LibreO...
mohanapriyanarayanan@localhost:~
Home Page - Title - Mozilla Firefox
1 / 4

Workflow Label	Submit Host	Submit Directory	State	Submitted On
merge	localhost.localdomain	/home/mohanapriyanarayanan/merge/submit/mohanapriyanarayanan/pegasus/merge/run0001	Running	Wed, 11 May 2016 17:47:57
pipeline	localhost.localdomain	/home/mohanapriyanarayanan/pipeline/submit/mohanapriyanarayanan/pegasus/pipeline/run0001	Successful	Wed, 11 May 2016 10:42:18
split	localhost.localdomain	/home/mohanapriyanarayanan/split/submit/mohanapriyanarayanan/pegasus/split/run0001	Successful	Wed, 11 May 2016 10:22:18

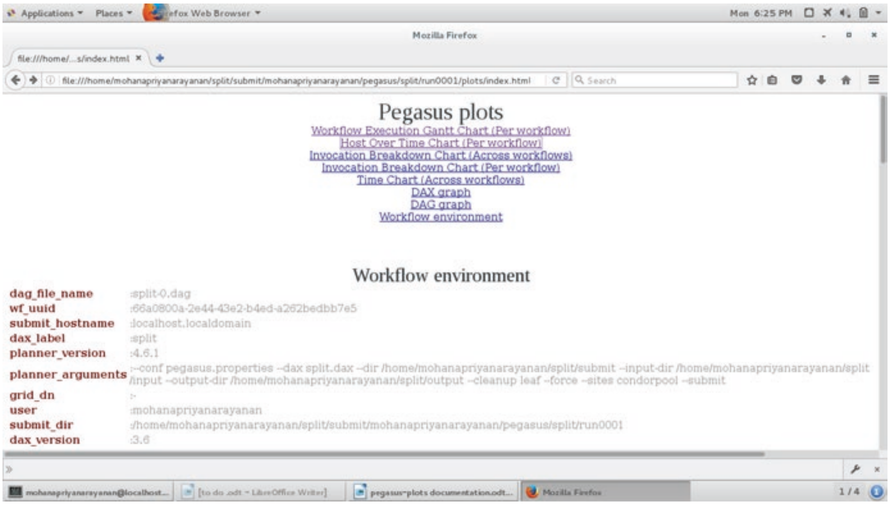
Showing 1 to 3 of 3 entries

Copyright © 2015 University of Southern California

Fig. 6.14 Status of the workflow execution in Pegasus WMS

to generate graphs and charts to visualize the workflow run. Pegasus-plots is a command to generate graphs and charts. This is placed in the plots folder of workflow directory. Figure 6.15 represents the Pegasus-plots generated for the spilt workflow in Pegasus WMS.

The combination of infrastructure as a service and platform as a service over the cloud forms an absolute workflow scheduling architecture, and it introduces various scheduling challenges. This section addresses such issues of scheduling workflows in cloud. While proposing a new scheduling algorithm, certain issues are to be


Applications
Places
Firefox Web Browser
Mon 6:25 PM
Mozilla Firefox
file:///home/...s/index.html
file:///home/mohanapriyanarayanan/split/submit/mohanapriyanarayanan/pegasus/split/run0001/plots/index.html
Search
mohanapriyanarayanan@localhost...
[to do .odt - LibreOffice Writer]
pegasus-plots documentation.odt...
Mozilla Firefox
1 / 4

Pegasus plots

Workflow Execution Gantt Chart (Per workflow)
Host Over Time Chart (Per workflow)
Invocation Breakdown Chart (Across workflows)
Invocation Breakdown Chart (Per workflow)
Time Chart (Across workflows)
DAX graph
DAG graph
Workflow environment

Workflow environment

dag_file_name	:split-0.dag
wf_uuid	:66a0800a-2e44-43e2-b4ed-a262bedbb7e5
submit_hostname	:localhost.localdomain
dax_label	:split
planner_version	:4.6.1
planner_arguments	:--conf pegasus.properties --dax split.dax --dir /home/mohanapriyanarayanan/split/submit --input-dir /home/mohanapriyanarayanan/split /input --output-dir /home/mohanapriyanarayanan/split/output --cleanup leaf --force --sites condorpool --submit
grid_dn	:-
user	:mohanapriyanarayanan
submit_dir	:/home/mohanapriyanarayanan/split/submit/mohanapriyanarayanan/pegasus/split/run0001
dax_version	:3.6

Fig. 6.15 Charts generated

considered for designing an efficient algorithm. Some of the issues of scheduling workflows in cloud are (Mohanapriya et al.) [14]:

- Scheduling algorithms for grids and clusters are focused on deadlines or reducing the make span without cost consideration. For cloud, scheduling algorithms need to be developed by considering the cost as an important factor to avoid cost overheads.
- Adaptive nature: cloud is a dynamic environment with a variation of performance during the workflow execution; therefore an adaptive scheduling solution is required.
- Migration: migration of jobs for load balancing also affects the performance of workflow scheduling.
- Virtual machine replacement: workflow scheduling algorithm should efficiently replace the VM in case of failure.
- There are two main stages in cloud environment before workflow execution: resource provisioning and task-resource mapping. Grid and cluster environments deal only with task-resource mapping stage since they are static environments whose configurations are already known. A major problem of provisioning resource is to predict the amount and resource type that a workflow application would request, which affects the cost and make span of a workflow.
- Virtual machines offered by current cloud infrastructure are not exhibiting stable performance [15]. This has a considerable impact on a scheduling policy. Boot time and stop time of the VM are the other essential factors that should be considered for cloud scheduling [11].

- Integrated architecture: the first challenge is the integration of workflow management system with the cloud. A workflow engine should be designed with a strong functionality to deal with large-scale tasks.
- Large-scale data management and workflow scheduling: scientific workflow applications are more data intensive. Data resource management and data transfer between the storage and computing resources are the main bottleneck. It is very important to find an efficient way to manage data needed by the workflows.

6.5 Conclusion

This chapter describes the deployment of Pegasus Workflow Management System in the OpenStack cloud and the detailed description of the creation and execution of workflows. A comparison between executing a workflow in standalone machine and on different OpenStack instances is also included. Based on the results of the comparison, it records the inference that cloud execution provides better results. The scheduling algorithm in Pegasus along with the various issues in scheduling workflows in the cloud has also been detailed. These issues are to be taken care of while scheduling workflows in a cloud environment. The chapter provides a well-defined description about the workflow management system that would aid the researchers to carry out their work in the field of workflow management and scheduling in cloud. In this work a consideration of executing workflows in the condorpool is addressed.

References

1. Mao M, Humphrey M (2011) Auto-scaling to minimize cost and meet application deadlines in cloud workflows. In: High-performance computing, networking, storage and analysis (SC), International conference, pp 1–12, 12–18 Nov, 2011
2. Juve G, Chervenak A, Deelman E, Bharathi S, Mehta G, Vahi K (2013) Characterizing and profiling scientific workflows. Futur Gener Comput Syst 29(3):682–692
3. Prajapati HB, Shah VA (2014) Scheduling in grid computing environment 2014 fourth international conference on advanced computing & communication technologies, Rohtak, pp 315–324. doi:10.1109/ACCT. 2014.32
4. Wu F, Wu Q, Tan Y (2015) Workflow scheduling in cloud: a survey. J Supercomput 71(9):3373–3418
5. Yu J, Buyya R (2005) A taxonomy of scientific workflow systems for grid computing. ACM SIGMOD 34(3):44–49
6. Masdari M, ValiKardan S, Shahi Z, Azar SI (2016) Towards workflow scheduling in cloud computing: a comprehensive analysis. J Netw Comput Appl 66:64–82
7. Michael RG, Johnson DS (1979) Computers and intractability: a guide to the theory of NP Completeness. WH Freeman Co., San Francisco
8. Armbrust M et al (2009) Above the clouds: a Berkeley view of cloud computing, white paper, UC Berkeley

9. Foster I et al (2008) Cloud computing and grid computing 360-degree compared. Grid computing environments workshop (GCE ‘08)
10. Peter M, Grance T. The NIST definition of cloud computing. NIST Special Publication. http://dx.doi.org/10.6028/NIST.SP.800-145
11. Rodriguez MA, Buyya R (2014) Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. IEEE Trans Cloud Comput 2:222–235
12. https://pegasus.isi.edu/overview/
13. Thain D, Tannenbaum T, Livny M (2005) Distributed computing in practice: the condor experience. Concurr Comput Exper Pract 17(2–4):323–356. http://dx.doi.org/10.1002/cpe.v17:2/4
14. Mohanapriya N, Kousalya G, Balakrishnan P (2016) Cloud workflow scheduling algorithms: a survey. Int J Adv Eng Technol VII(III):88–195
15. Iosup A, Simon O, Nezih Yigitbasi M, Prodan R, Fahringer T, Epema DHJ (2011) Performance analysis of cloud computing services for many-tasks scientific computing. IEEE Trans Parallel Distrib Syst 22(6):931–945

Chapter 7
Workflow Predictions Through Operational Analytics and Machine Learning

Abstract Workflow execution employs predictive analytics to extract significant, unidentified as well as precious insights from several stages of execution. Further, the operational analytics integrates these valuable insights directly into decision engine which enables analytical as well as machine learning-driven decision-making for an efficient workflow execution. This chapter highlights several analytical and machine learning approaches that are practiced in workflow predictions. Additionally, it explains the significance of hybrid approach which includes both analytical and machine learning models for workflow prediction. Finally, it describes the hybrid approach employed in PANORAMA architecture using two workflow applications.

7.1 Introduction

Workflow prediction is an act of forecasting the execution time and cost profile for each task on top of the workflow as a whole. Conversely, the execution time and cost primarily depend on several factors such as: workflow structure, nature of the application and its execution environments like compute and storage resources and their network connectivity. Obviously, these factors introduce a lot of complexities in deciding from which cloud provider, what type of cloud instance profile, and how many such instances need to be leased for executing a given workflow. There are two approaches proposed in the literature to handle the abovementioned scenario: analytical models (AMs) and machine learning models (MLMs). This chapter emphasizes the fundamentals of AM and MLM along with their limitations. Following that, it explains a hybrid model to reap out the benefits of both AM and ML approaches.

Section 7.2 discusses about different workflow prediction approaches. Important concepts, such as the need of workflow prediction models along with the associated challenges, merits as well as demerits of AM, MLM, and hybrid models, are explained in this section.

Section 7.3 highlights various AM-based prediction approaches available in the literature. Section 7.4 elucidates different MLM-based prediction approaches along with their pros and cons. Section 7.5 points out the hybrid approach for performance prediction.

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_7

Apart from this, it also explicates the hybrid approach using PANORAMA as a case study in Sect. 7.6. Additionally, the architecture and operation of PANORAMA are explained in detail using two workflow applications.

Finally, this chapter is closed off with several concluding remarks.

7.2 Workflow Prediction

Significant advancements in scientific computing lead to complex applications that require huge amount of computational power and storage capabilities. Most of these applications are expressed as scientific workflows that include several data- or control-dependent computational as well as data manipulation steps. In these workflows, each step may require different software and hardware requirements to process the supplied data. Hence, the real challenge relies on task scheduling, maintaining inter-task dependencies, and staging of datasets to/from executing resources. Traditional expensive high-performance systems have limited amount of resources and are not capable of creating scalable dynamic execution environments. However, cloud computing, an on-demand environment with unbounded number of heterogeneous resources, is a potential candidate to provide resources for scientific workflows. But, the varying demands and nature of each step in complex scientific workflows introduce a lot of challenges in deciding an appropriate cloud provider, number and characteristics of computing, or storage resources required to complete its execution. Hence, the planning and execution of workflows require advance prediction of both execution time and cost profile for each step/task/activity as well as the workflow on the whole. However, the execution time and cost of the workflow depend on several attributes, such as workflow structural properties, application properties, execution environment properties, and resource and network properties. In concise, the time as well as cost-efficient workflow execution demands an application-specific performance prediction system (APPS) which predicts the specific application's resource requirements (such as cloud instance hardware profile and their counts) using the fine-grained application-specific profile information (like runtime, CPU load, memory usage, storage, and bandwidth usage). With this objective, several performance prediction systems for grid computing are proposed in the literature. However, these systems can be modified to meet out the performance prediction requirements of cloud computing. For example, the approach proposed by Kurowski et al. [1] can be directly associated with the hardware profiles of cloud instances which in turn provides the cost of execution in cloud environment.

7.2.1 Challenges in Designing an APPS

Firstly, the diverse nature of cloud applications (sequential, distributed, batch or interactive, singular, periodic, or data centric) and their assorted executing resources (parallel, homogenous, or heterogeneous cluster) lead to heterogeneous cloud data. For instance, the input or output data for a high-throughput computing (HTC) application executed over heterogeneous cluster is different from high-performance computing (HPC) application executed over homogenous cluster. Secondly, the scientific workflows comprise of multiple steps/tasks/jobs where each step requires different software, hardware, and QoS requirements. This demands the fine-grained profiling of each step to classify them into compute-, memory-, disk-, or I/O--intensive category. Obviously, this classification will be helpful to optimally choose the type and counts of cloud instances for that particular step. For example, if a workflow step is categorized as compute-intensive class by analyzing its CPU, RAM, disk, and I/O usage, then choose the compute-intensive cloud instances for that step than the memory- or I/O-intensive cloud instances. Apart from this, one can easily forecast the cost involved in executing the whole workflow by specifying the deadline of execution. In short, to successfully design an APPS, it is mandatory to address the problems from several dimensions which are listed as follows:

1. Capable to gather several static and dynamic parameters of each application or a workflow step, such as execution time, average/min/max CPU and RAM usage, CPU load, disk read/write time, I/O wait time, size of the input or output files, and MPI communication time.
2. Capable to categorize each workflow step into compute-, memory-, disk-, or I/O--intensive class. Besides, it should have the ability to identify the resource consumption similarity among the workflow steps.
3. Capable to extract the relationship between the execution time and resource configuration (like CPU, RAM, storage) by analyzing the parameters gathered in step 1. Besides, it should be equipped with the facility to forecast either the execution time if the resource configuration is given or vice versa.
4. Capable to forecast the possible errors in prediction in order to facilitate the statistical corrections to the predictions.

7.2.2 Workflow Prediction Approaches

As mentioned earlier in Sect. 7.2.1, different research communities proposed several performance prediction approaches that lead to different models which are used to improvise the efficacy of scheduling in cloud environments. However, these approaches can be classified into two broad categories: analytical model (AM) and machine learning model (MLM) approaches.

AM is a reference technique to conduct the performance estimation and prediction of computing platforms for different application contexts. It gathers the internal characteristics of target systems or applications and transforms such knowledge into a comprehensive mathematical model which represents the association of performance with the parameters included in the model. Typically, AM necessitates no or little training in order to predict the future performance of a target scenario with good overall accuracy. However, the accuracy is seriously affected in scenarios which do not hold the model's assumptions. Besides, the increasing computer architecture complexities and the multi-tenant cloud environments which hide the infrastructure details for their applications hinder the development of accurate AM models. In contrast, MLM utilizes the actual behavior of the system/applications in different settings to extract the statistical behavioral model of performance and its influencing characteristics. However, the prediction accuracy of MLM models majorly depends on the features' space of the dataset used in the initial training phase. For example, predictions which target the scenarios with more feature space dimensionality than the training dataset obviously have poor accuracy. Further, it is infeasible to conduct the comprehensive training for all possible input configurations.

The MLM-based prediction approaches can be further classified into spatial-temporal correlation models (STCs) and independent data tuples (IDTs). The STC models are evolved, while the data are collected over both in time and space. For instance, the weather data are collected at uniform intervals from different places. Thus, the STC models analyze the time-series data by considering both temporal and spatial correlations. The primary challenge in this model is how to handle the missing data in the time series. Besides, the STC models can be further classified into univariate and multivariate time-series models. The univariate time-series models analyze the data of single variable which are collected in time order. In contrast, the multivariate time-series models extract the knowledge from time order data points of several variables.

On the other hand, IDT models define templates of attributes to classify the jobs and then apply the statistical approaches to generate predictions. Further, IDT models are divided into artificial intelligence (AI) models and data mining models. The AI models apply the artificial intelligence techniques on the historic information to categorize the workloads. Such categories are stored into the knowledge base as rules to produce the predictions for future jobs. Similarly, the data mining models apply the statistical methods on historic information to identify the similar datasets. After that, the instance-based learning (IBL) techniques are applied to predict the execution times.

7.3 AM-Based Performance Prediction Systems

Performance (PACE) [2] analysis environment is a toolkit to evaluate and predict the performance as well as bottlenecks of several applications on different parallel platforms. It characterizes the whole system into three layers: software layer, parallelization layer, and hardware layer. The evaluation engine of PACE toolkit gathers the information from these layers and simulates the applications at a faster time scale to generate the estimation of elapsed time, scalability, and resource usage. Here, the elapsed time represents the predicted application runtime under a given application and system parameters. The scalability parameter expresses the performance changes in application while varying the application (like input data size) and system parameters (like number of processors).

Task profiling model (TPM) [3] is an APPS which predicts the runtime and load profile of job tasks in a standard time-sharing system. In such systems, the variations in CPU load seriously affect the execution time of CPU-intensive applications. Hence, the TPM initially gathers the load of each user in the system, and then it collects the free load profile (FLP) which is a load profile of each task with no background load. Further, it utilizes these information to predict the load profile of future jobs, thereby scheduling them to suitable computing resources.

DIMEMAS [4] is a simulator that predicts the performance of an MPI application using its execution trace file that contains the computation/communication pattern along with the configuration file which models the target architecture.

LaPIe [5] is a generic framework that identifies an efficient execution plan by associating the best communication strategy-schedule scheme which minimizes the completion time of a parallel application by considering the overall collective communication time. For that, it divides the entire network into logical clusters, generates performance models for several communication schemes, and chooses the best scheme for each logical cluster. The entire framework is modeled as pLogP model that contains communication latency (L), message gap between messages of size “m,” and the number of processes (P).

The performance prophet of Askalon [6] predicts the performance of distributed/parallel applications using Teuta and Performance Estimator components. Using Teuta, the user represents the application as UML models and embeds the performance and control flow information. These models are converted into an intermediate form and are stored in the data repository. Later, performance predictor reads these models and estimates the application performance on selected platforms. If the performance is not satisfied, the user may alter the application model or opt for an alternate target platform till the user gets satisfied. In contrast, G-Prophet uses the historic runtimes and input size of each task to predict the runtime of future jobs. If the future jobs with the same input size are executed over a similar machine, then the runtime is predicted using the load and memory. If not, firstly calculate the new runtime for this new machine, and then the future runtime is predicted using the load and memory.

The GAMMA [7] model predicts the relationship of a parallel application to the cluster architecture, thereby mapping a suitable cluster to that application. For that, it models the parallel application as a ratio of the number of operations (O in Mflops) to be executed on a single processor to the amount of data to be transferred (S in Mwords) from one processor to another. Similarly, it models the cluster architecture as a ratio of the number of operations (Mflops/s) that the processor can perform while sending a single word from one to another (Mwords/s). A parallel machine has the capability to perfectly execute an application, if it meets the inequality << which ensures that the communication is inferior to computation.

Performance skeleton is a scaled-down version of an application that is created from its execution traces and is executed over the target node to predict its runtime. This approach follows three steps to predict the execution time:

1. Firstly, the application is executed on a test-bed and its execution traces are collected
2. Secondly, extract the execution signature from the traces by compressing the recurring patterns.
3. Thirdly, devise a scaled-down version of an application called skeleton from the execution signature.
4. Finally, these skeletons are executed over the target node for which the runtime needs to be estimated.

Framework for rapid implementation of data mining engines in grid (FREERIDE-G) [8] middleware enables the development of grid-based data mining applications that utilizes the data from multiple remote storage repositories. Besides, it also proposes a prediction model that selects a suitable replica set and the computing node for processing by forecasting the data retrieval, communication, and processing times.

Fast Agent's System Timer (FAST) [9] is an API which dynamically forecasts the memory usage, computation, and communication times of a given application executed over network-enabled servers. It has two data acquisition modules: static and dynamic. The former module characterizes the time as well as space requirements of an application, whereas the latter module forecasts the dynamic characteristics of resources. Nevertheless, the former uses the Lightweight Directory Access Protocol (LDAP) to read or search static data, while the latter uses the Network Weather Service (NWS) tool to dynamically monitor both host and network parameters. After gathering static and dynamic data, the FAST API uses four important functions to forecast memory, computation, and communication time:

1. fast_comm_time(data_desc, source, dest) – it estimates the time to transfer the data from source node to destination node where actually the computation is going to be done by considering the current network condition.
2. fast_comp_time(host, problem, data_desc) – it estimates the time for executing the problem on a host for a given data by considering the theoretical as well as actual load of the host.
3. fast_comp_size(host, problem, data_desc) – it estimates the memory space needed for the problem.
4. fast_get_time(host, problem, data_desc, localization) – it aggregates the outputs from other functions to forecast the runtime time of a problem on a specified host for a given data.

Carvalho et al. [10] proposed a prediction model which forecasts the quantity of resource available for a peer in a desktop grid environment. The idea is whenever a consumer peer submits tasks to a desktop grid, it collects the resource information from potential donor peers and predicts the quantity of resources available for consumer peer in a future time. This prediction phase should be initialized just before submitting the tasks. The proposed model is validated by a trace-driven simulation, and prediction error is calculated for each donor peer by subtracting the ratio of estimated requested (ER) resources from obtained requested (OR) resources.

7.4 MLM-Based Performance Prediction Systems

Network Weather Service (NWS) [11] is a system to generate a short-term forecast about the host as well as network parameters. To achieve this, it has several software sensors (CPU sensors to measure CPU availability and network sensors to gather the bandwidth and end-to-end round-trip latency) to periodically measure the network- as well as processor-related parameters and send it to persistent state managers. These time-stamped series data are fed to forecaster, where a set of forecasting models are applied to select the most accurate forecasting techniques which has the lowest prediction error. The forecaster uses moving average (MA), auto regressive (AR), and sliding window or adaptive window average techniques to predict the CPU availability and TCP end-to-end throughput and availability.

Dinda [12] conducted a comprehensive statistical analysis on the host loads which are collected from wide varieties of resources ranging from single PC to clusters. The important conclusions of this statistical analysis are:

1. Host load varies in complex ways and exhibits epochal behavior inherently. Alternatively, it shows a constant load changing pattern over a long duration of time.
2. Due to the presence of epochs in historic load, it is feasible to model the future loads. However, the historic loads do not expose seasonality.
3. Additionally, the load traces exhibit heavy self-similarity for a Hurst parameter values from 0.63 to 0.97.

Though it is so hard to model and predict the load since it is varying in complex ways, Dinda proved that by applying a time-series analysis methodologies, like autocorrelation or periodogram on historic load, it is feasible to predict the future load values from past values.

Mostly, the traditional workload models conduct statistical analysis which uses probability distribution function to extract the static features from the workload traces. However, Song et al. [13] proposed a workload model that uses Markov chains to model the job parameters by considering their static as well as temporal relationships. Here, an independent Markov chain is created for each job parameter. Later, all are combined to utilize the sequential correlation of the job to model the future job.

Yang et al. [14] proposed one-step-ahead CPU load predictions using weighted sliding window approach that gives more weight to the most recent historic time-series data. In this context, two different approaches, namely, homeostatic and tendency-based predictions, were proposed. In homeostatic, if the current CPU load is greater or lesser than the mean of historic values, then the next value may be decreased/increased. In contrast, the tendency-based approach uses the trend (increase or decrease) of historic CPU load to forecast the future CPU load. Later, the model proposed in [15] was enhanced to predict both mean and variance of CPU load in the near future. Further, multi-source parallel data transfer enables the transmission of data from multiple replica sites, thereby reducing the data access time. Indeed, the real challenge lies in selecting which data partition from which replica site. Yang et al. [16] proposed a methodology to forecast the mean and variance of future network capabilities to estimate the quantity of data to be transferred from each replica site by enhancing the predictors of NWS. After that, effective bandwidth is calculated for each channel using this predicted mean and variance, which in turn used to calculate the time needed to transfer the data from each site.

The traditional point value auto regression (AR) method obtains the historical data which is available in a fixed interval as an input to predict the future workload. Intuitively, it may not be suitable for long running jobs as its interval does not cover the entire load variations. Wu et al. [17] proposed an adaptive hybrid model (AHM) which expands the point AR to a confidence window, where window intervals can be dynamically altered based on load changes. Further, it utilizes the Savitzky-Golay filter to smooth out load variations and Kalman filter to reduce the errors, thereby increasing the prediction accuracy.

Basically, applications of the same kind may have the same runtimes than applications which don't have anything in common. Smith et al. [18] proposed a methodology to predict the application runtimes using the historical data of past similar runs. Firstly, it defines a set of templates with different characteristics and categorizes the applications based on the similarity with the template characteristics. Here, if two applications fall into the same category, they will be treated as similar applications. Then the runtime is predicted using the mean or regression predictor from the history of similar past runs. Similarly, IBL-based runtime prediction approaches are explained in [19]. Firstly, it classifies the history of previous runs using the k-nearest neighbor (KNN) and genetic search along with several distance functions [20], and then it predicts the runtimes using IBL techniques. Likewise, eNANOS [21] is yet another approach to predict runtime and memory of applications from historical information using statistical predictors, like mean, median, standard deviation, and linear regression. In addition, a framework [22] is proposed which uses data mining as well as artificial intelligence techniques to predict the availability, CPU, and memory loads in a highly volatile environment. Moreover, the individual application's resource usage is an important piece of information to job schedulers to achieve better resource utilization. Hence, to predict the resource utilization of applications, a binary tree-based classification algorithm called predicting query runtime (PQR) is extended (PQR2 [23]) to accommodate multiple regression methods in the leaves of a tree, thereby identifying a suitable regression method. The efficacy of PQR2 is validated on two bioinformatics applications BLAST and RAxML. The approach proposed in [24] firstly classifies the history of jobs into small, medium, and big category; secondly, it identifies the critical parameters to predict the runtime and finally applies the genetic algorithm to estimate these parameters. A generic Bayesian-neural network approach to accurately predict the runtime of any scientific application is explained in [25]. The Bayesian network dynamically identifies the critical factors that affect the performance and fed them to radial basis function neural network (RBF-NN) for more accurate runtime predictions.

7.5 Hybrid Performance Prediction Systems (HPPS)

In concise, pure AM or MLM approaches are not appropriate for designing a robust APPS that predicts the optimal resource configuration for their applications which gives the assured performance. To get the best of these two approaches, a new hybrid APPS with an ensemble of both AM and MLM approaches is explained in this section. This hybrid APPS starts with a pre-built AM which is incarnated with lesser training time than its MLM-based predictors. However, the MLM component improvises the accuracy of hybrid APPS as the new data arrive from operational systems.

7.5.1 Opportunities and Challenges for HPPS

During their training phase, any ML algorithm develops a model by utilizing the features' space of input values and their corresponding output values from their training data. More precisely, the model takes any input which is not available in the training set and estimates its corresponding output value. Obviously, there is a difference between the actual and estimated value which is known as an error. These errors can be minimized by tuning the model using some statistical approaches which actually tries to reduce the errors during the training itself. Further, the model can be updated every time new observations are included in the training data. In contrast, an AM is a static or immutable model built using a priori knowledge about the system and cannot be updated or does not involve any training. However, the prediction accuracy of AM depends on several internal parameters which can be tuned to improvise the efficacy of the model. This dependency provides an excellent opportunity to integrate MLM into AM. Firstly, a MLM is developed using the samples of target system which are collected over a period of time to estimate these internal parameters. Secondly, these estimated values are fed into AM to adjust its predictions. In concise, the internal parameters of AM are modified dynamically whenever the MLM is updated as new data arrives. The primary benefits of exploiting the synergies between AM and MLM are to realize a more accurate HPPS than their individual implementations.

However, the major challenge is as follows: How to combine AM and MLM?

Here, three different approaches are proposed to combine AM and MLM.

1. Single MLM Single AM (SMSA)
2. Multiple MLM Single AM (MMSA)
3. Multiple MLM Multiple AM (MMMA)

Single MLM Single AM Here, AM is used to instantiate a performance model for a target system instantly without any training. However, it depends on numerous internal parameters to adjust the model's prediction. For instance, in an adapted AM uses an inferior set of internal parameters to estimate the model's output, it naturally leads to an error in prediction. Now, MLM is used to learn the influencing candidate internal parameters and is then fed into AM to dynamically adjust the prediction, thereby improvising the accuracy. During learning, the MLM may use either whole region or exclusively on the regions of feature space of internal parameters which leads to maximum prediction error on AM.

Multiple MLM Single AM This approach is similar to SMSA except that multiple MLMs are independently used to learn the feature space of internal parameters. Obviously, each MLM feeds the same AM which in turn produces their corresponding prediction errors. The MLM-AM combination with minimum prediction error is used to predict the model's output.

Multiple MLM Multiple AM MMMA is analogous to MMSA wherein multiple AMs are used instead of single AM. Each MLM learns as well as feeds all the AMs and measures their respective errors for each AM. The MLM-AM combination with minimum error is chosen to predict the model's output.

7.6 Case Study

Scientific workflows are practiced in numerous science domains, like bioinformatics, chemical modeling, climate modeling, earth science, and many others. These scientific workflows articulate complex applications as several data and control-dependent computational as well as data manipulation steps. However, each step may require different software and hardware to process the supplied data. Besides, these steps may access data from several repositories as files. Consequently, the performance of workflow not only depends on the performance of computational tasks but also depends on the performance of storage and network devices. But the state-of-the-art workflow scheduling approaches mostly concentrate on resource provisioning for computational task and pay less attention to data access as well as data movement. Unfortunately, current workflow monitoring systems are also biased toward this theory and monitor the task execution ignoring the data access patterns. Hence, to develop a predictive model for workflow applications, it is mandatory to monitor their execution data and correlate it with the infrastructure. PANORAMA [26] is one such project which tries to model and predict the performance of extreme-scale workflows funded by the US Department of Energy (DoE). This section illustrates the architecture of PANORAMA along with the detailed

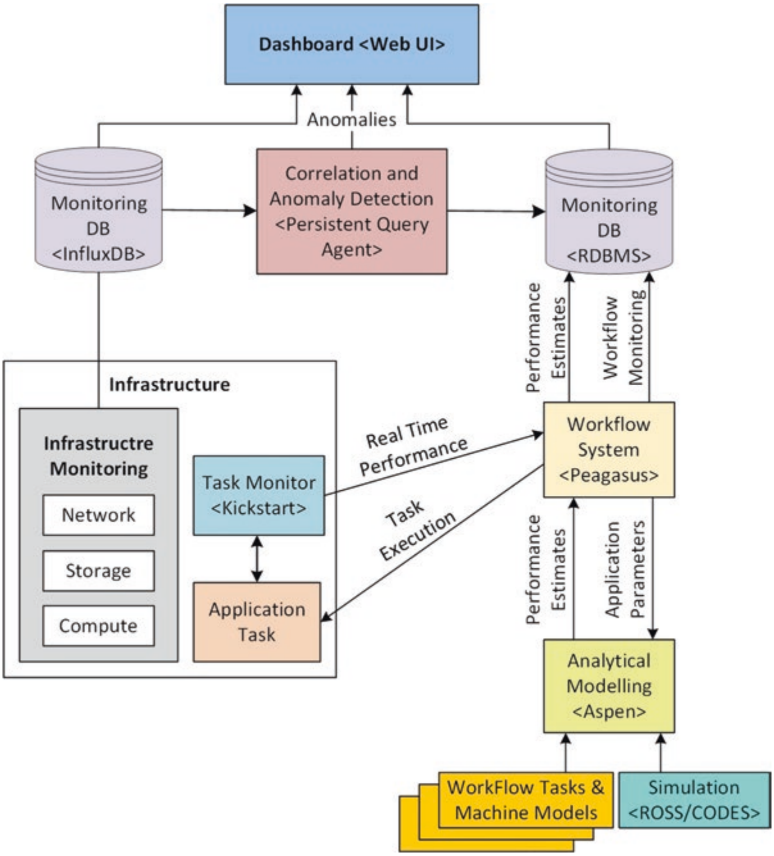


Fig. 7.1 PANORAMA architecture

discussion on techniques for diagnostic monitoring and prediction of workflow applications.

7.6.1 *PANORAMA: An Approach to Performance Modeling and Diagnosis of Extreme-Scale Workflows*

The PANORAMA architecture (refer to Fig. 7.1) has the following components.

- Pegasus Workflow Management System (WMS)
- Aspen Analytical Modeling
- Rensselaer’s Optimistic Simulation System (ROSS)
- Co-Design of Exascale Storage (CODES) Simulation
- Workflow, Task, and Machine Models

- Workflow, Task, and Infrastructure Monitor
- Workflow Database
- Monitoring Database
- Persistent Query Agent
- Dashboard

Pegasus Workflow Management System Pegasus WMS creates an abstract workflow or workflow instance (i.e., DAX) which clearly expresses the methodology to conduct the analysis at the application level, without revealing the details of execution environment. Alternatively, the abstract workflow decouples workflow description from its target executing resource requirements. This interesting feature enables the scheduler to generate performance- or reliability-guided execution plans which can be ported among several execution environments. During planning phase, it invokes Aspen tools to forecast the performance parameters for tasks/workflow using the supplied application parameters. Consequently, Aspen uses simulation (i.e., ROSS/CODES for network and storage) and analytical models (i.e., workflow, task, and machine) to produce performance predictions. From these predictions, Pegasus clearly identifies both the input data repositories and executing resource descriptions and maps them to executing resources. Further, the fault-tolerance mechanism of Pegasus supports both rescheduling and check pointing at either task or workflow level. The rescheduling facility tries to recover the errors by retrying faulty tasks or entire workflow. Also, it supports the replanning of either the entire workflow or part of workflow whose execution state is obtained from check-pointing mechanism.

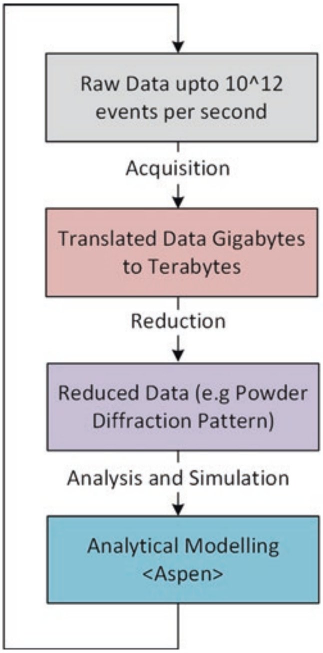
Aspen Analytical Model Aspen is a modular, reusable, and extensible performance modeling system. Its domain-specific language (DSL) uses the information obtained from functional simulation to develop structured analytical models for monolithic science applications and high-performance computing (HPC) architectures. However, the traditional Aspen is extended to support the development of analytical models for scientific workflow scenarios. With this model, Aspen has the capability to explicitly trace algorithmic information, control, and dataflow of workflows. PANORAMA uses Kickstart as a workflow monitoring system to collect the profile data during execution which contains I/O, memory, CPU usage, and runtime data. Using these information, a history-based performance model is developed for a workflow which can be used to produce estimates for cores and memory capacity of computing resources, network data transfer throughput, and storage system performance.

ROSS and CODES Framework ROSS is a highly scalable simulation framework which models the network as parallel discrete event simulations. Similarly, CODES simulation engine models the storage devices, network, software, and I/O as parallel discrete event simulation. The Aspen represents the workflow as a high level analytical model and uses the simulation for the interaction of storage or network. For instance, the Aspen generates I/O requests and sent it to CODES simulator to access a storage system, whereas the rest of the parts uses the analytical model.

Workflow, Task, Application, and Infrastructure Monitoring Workflow characterization and analysis demand precise runtime traces and profile data to model the system behavior, anomaly detection, and diagnosis. Workflow monitoring gathers the event traces during the execution of workflows which are analyzed to detect its behavior. The STAMPEDE framework is used to extract the events such as submission time, start time, failure report time, and finish time by analyzing the log files. These events can be used to calculate the runtimes as well as queue wait times. In contrast, the task monitoring gathers the runtime parameters of each task in a workflow using Kickstart monitoring tool of STAMPEDE framework. It collects the CPU and memory usage, I/O statistics, and file access patterns of each task which will be used to develop a machine learning model that predicts the workflow task's runtime behavior. This model is integrated with Monitor-Analyze-Plan-Execute (MAPE-K) to support online task estimation as the workflow execution progresses. The application monitoring gathers the performance-related metadata, such as the number of steps completed per hour, error messages, and progress percentage which can only be collected during its execution. Finally, the infrastructure monitoring examines soft failures in networks, like network performance is just a fraction of its crest efficiency. It uses the perfSONAR to continuously monitor packet loss, network latency, and throughput. Besides network, it also monitors compute and storage resources which guide the scheduling algorithm while making decisions.

Anomaly Detection and Diagnosis In reality, the real-time execution behavior of workflows is very different from the predicted behavior of analytical model. These anomalies have to be detected at various stages of execution using either online or offline approaches. The offline approach categorizes the anomalies based on their severity, which is decided using the predefined threshold levels. Here, the infrastructure anomalies such as degraded I/O performance, disk or filesystem failure, network congestion, and packet loss severely affect the application performance. Additionally, the application anomalies like execution errors, input data unavailability and workflow anomalies, heavy task failures, data staging failures, and priority issues contribute much toward workflow performance degradation. Since the threshold levels are predetermined, offline methods do not include any additional overhead during workflow execution. However, the online approach uses Persistent Query Agent (PQA), which continuously executes queries over several data sources to compare the reference metrics with the actual data and generate notifications whenever anomalies occur. The reference values for application, workflow, and infrastructure need to be stored well in advance with which the PQA compares the actual data to generate events. For example, at any instant, the workflow performance is compared against several infrastructures monitoring information (i.e., CPU, RAM, network, and storage) from different systems to identify which systems hinder the performance. These reference values are decided either by the benchmark executions or from Aspen prediction model. Apart from this, it is easy to identify the persistent query which generates the events. From that persistent query, it is easy to identify the root cause (i.e., the parameter) for event by simple reverse engineering. Additionally, the PQA uses publish-subscribe mechanism while distributing notifications.

Fig. 7.2 SNS example workflow


Raw Data upto 10^12 events per second
Acquisition
Translated Data Gigabytes to Terabytes
Reduction
Reduced Data (e.g Powder Diffraction Pattern)
Analysis and Simulation
Analytical Modelling <Aspen>

Motivating Use Cases The PANORAMA framework uses the following applications for their modeling experiments: Spallation Neutron Source (SNS) (refer to Fig. 7.2) and Accelerated Climate Modeling for Energy (ACME) (refer to Fig. 7.3). SNS application contains two different workflows: post-processing workflows and pre-processing workflows. The post-processing workflows firstly conduct the SNS experiments and then process the resultant data, whereas the pre-processing workflows initially conduct the data analysis and simulation to guide the SNS experiments. The primary objective of the post-processing workflow is to produce a short proton sequence to hit a mercury target which generates neutrons by spallation. These scattered neutron events are captured by array of detectors in NOMAD. After a series of operations over these events, it is reduced to powder diffraction pattern over which data analysis is conducted to pull out information. In contrast, the preprocessing workflow tries to refine a target parameter headed for fitting experimental data. In this SNS refinement workflow (refer to Fig. 7.2), each set of parameters is given as input to series of parallel NAMD. The initial simulation estimates the equilibrium followed by production dynamics. Then the output is sent to AMBER and Sassena from where the final output is transferred to client desktop. For more information on SNS workflow, refer to [26]. The ACME application studies (refer to Fig. 7.3) about the climate change by integrating the models of ocean, land, atmosphere, and ice. However, each part of the workflow demands diverse software and hardware spread across several places in DoE labs. The primary objective of PANORAMA is to automate the monitoring, resubmission, and reporting in several stages of ACME application. Here, a huge simulation is sliced into different stages.

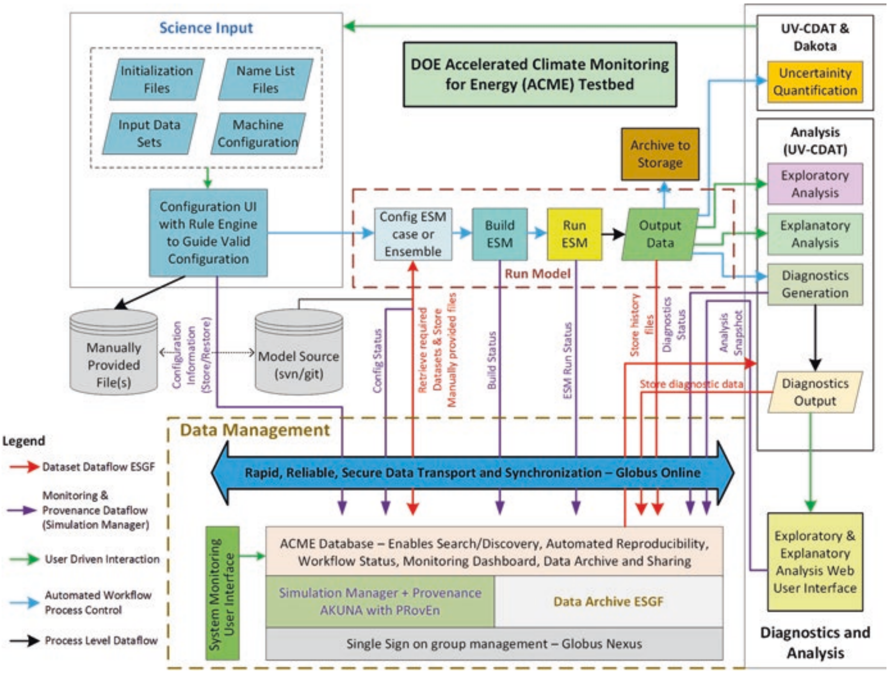

Science Input
Initialization Files
Name List Files
Input Data Sets
Machine Configuration
Configuration UI with Rule Engine to Guide Valid Configuration
DOE Accelerated Climate Monitoring for Energy (ACME) Testbed
Archive to Storage
Config ESM case or Ensemble
Build ESM
Run ESM
Output Data
Run Model
UV-CDAT & Dakota
Uncertainty Quantification
Analysis (UV-CDAT)
Exploratory Analysis
Explanatory Analysis
Diagnostics Generation
Diagnostics Output
Manually Provided File(s)
Configuration Information (Store/Restore)
Model Source (svn/git)
Config Status
Retrieve required Datasets & Store Manually provided files
Build Status
ESM Run Status
Store history files
Diagnostics Status
Analysis Snapshot
Store diagnostic data
Legend
Dataset Dataflow ESGF
Monitoring & Provenance Dataflow (Simulation Manager)
User Driven Interaction
Automated Workflow Process Control
Process Level Dataflow
Data Management
Rapid, Reliable, Secure Data Transport and Synchronization – Globus Online
System Monitoring User Interface
ACME Database – Enables Search/Discovery, Automated Reproducibility, Workflow Status, Monitoring Dashboard, Data Archive and Sharing
Simulation Manager + Provenance AKUNA with PRovEn
Data Archive ESGF
Single Sign on group management – Globus Nexus
Exploratory & Explanatory Analysis Web User Interface
Diagnostics and Analysis

Fig. 7.3 The complete Accelerated Climate Modeling for Energy (ACME) application

Each stage has to be completed in a stipulated deadline. At the end of each stage, two files are created: restart file and history files. The restart files are given as input to resume the simulations. Additionally, the summary of simulation at each stage is extracted from history files and stored in climatologies, which are used to verify the correctness of simulation. Finally, the history files as well as climatologies are stored in HPSS and CADES infrastructure for future analysis.

7.7 Conclusion

This chapter highlights the significance of workflow predictions together with the influencing parameters. Further, it explains about the two prominent approaches in workflow prediction: AM and MLM. Subsequently, several notable approaches that utilize either AM or MLM to predict one or more resource parameters during workflow execution are given in detail. Following that, a hybrid approach which integrates both AM and MLM is emphasized along with its merits. Finally, the most recent PANORAMA architecture is explicated in detail and also spotted the usage of hybrid approach for workflow execution prediction. Consequently, Chap. 8 outlines the opportunities and challenges during orchestration as well as integration of workflows.

References

1. Kurowski K, Oleksiak A, Nabrzyski J, Kwiecien A, Wojtkiewicz M, Dyczkowski M, Guim F, Corbalan J, Labarta J (2005) Multi-criteria grid resource management using performance prediction techniques. In: CoreGrid integration workshop, Pisa, Itally
2. Nudd GR, Kerbyson DJ, Panaefstathiou E, Perry SC, Harper JS, Wilcox DV (2000) Pace-A toolset for the performance prediction of parallel and distributed systems. High Perform Comput Appl 14:228–252
3. Seneviratne S, Levy D (2011) Task profiling model for load profile prediction. Futur Gener Comput Syst 27:245–255
4. Badia RM, Escale F, Gabriel E, Gimenez J, Keller R, Labarta J, Muller MS (2003) Performance prediction in a grid environment. In: Presented at the international conference on grid computing, first European across grids conference, Santiago de Compostela, Spain
5. Steffenel LA (2005) LaPIe: communications collectives Adaptées aux Grilles de Calcul. PhD, INPG, Grenoble, France
6. Fahringer T, Jugravu A, Pllana S, Prodan R, Seragiotto C Jr, Truong HL (2005) ASKALON: a tool set for cluster and grid computing. Concurr Comput Pract Exp 17:143–169
7. Gruber R, Volgers P, Vita AD, Stengel M, Tran TM (2003) Parameterisation to tailor commodity clusters to applications. Futur Gener Comput Syst 19:111–120
8. Glimcher L, Agrawal G (2007) A performance prediction framework for grid-based data mining application. In: Presented at the International Parallel and Distributed Processing Symposium (IPDPS)
9. Desprez F, Quinson M, Suter F (2002) Dynamic performance forecasting for network-enabled servers in a heterogeneous environment. In: Presented at the international conference on Parallel & Distributed Processing Techniques & Applications (PDPTA), Las Vegas, USA
10. Carvalho M, Miceli R, Maciel Jr. PD, Brasileiro F, Lopes R (2010) Predicting the quality of service of a peer-to-peer desktop grid. In: 10th IEEE/ACM international conference on Cluster, Cloud and Grid Computing (CCGrid), Melbourne, Australia, pp 649–654
11. Wolski R, Spring N, Hayes J (1999) The Network weather service: a distributed resource performance forecasting service for metacomputing. Futur Gener Comput Syst 15:757–768
12. Dinda PA (2000) Resource signal prediction and its application to real time scheduling advisors. PhD, School of Computer Science, Carnegie Mellon University, USA
13. Song B, Ernemann C, Yahyapour R (2004) Parallel computer workload modelling with Markov Chains. In: Presented at the international conference on job scheduling strategies for parallel processing, NY, USA
14. Yang L, Schopf J M, Foster I (2003) Homeostatic and tendency-based CPU load predictions. In: 17th International Parallel and Distributed Processing Symposium (IPDPS 2003), Los Alamitos, California, USA, IEEE CD-ROM, p 9
15. Yang L, Schopf J M, Foster I (2003) Conservative scheduling: using predicted variance to improve scheduling decisions in dynamic environments. In ACM/IEEE international conference on Supercomputing (SC2003), Phoenix, Arizona, USA, pp 31–37
16. Yang L, Schopf JM, Foster I (2005) Improving parallel data transfer times using predicted variances in shared networks. In: Presented at the fifth IEEE international symposium on Cluster Computing and the Grid (CCGRID05), Washington, DC, USA
17. Wu Y, Hwang K, Yuan Y, Zheng W (2010) Adaptive workload prediction of grid performance in confidence windows. IEEE Trans Parallel Distrib Syst 21:925–938
18. Smith W, Taylor V, Foster I (1999) Using runtime predictions to estimate Queue wait times and improve Scheduler performance. In: International workshop on job scheduling strategies for parallel processing, San Juan, Puerto Rico, pp 202–219
19. Atkeson CG, Moore AW, Schaal SA (1997) Locally weighted learning. Artif Intell Rev 11:11–73
20. Wilson DR, Martinez TR (1997) Improved heterogeneous distance function. Artif Intell Res 6:1–34

21. Rodero I, Guim F, Corbalán J, Labarta J (2005) eNANOS: coordinated scheduling in grid environments. In: Presented at the parallel computing: current & future issues of high-end computing, Parco 2005
22. Andrzejak A, Domingues P, Silva L (2006) Predicting machine availabilities in desktop pools. In: IEEE/IFIP Network Operations & Management Symposium (NOMS 2006), Vancouver, Canada
23. Matsunaga A, Fortes JAB (2010) On the use of machine learning to predict the time and resources consumed by applications. In: Presented at the 10th IEEE/ACM international conference on Cluster, Cloud and Grid Computing (CCGrid), Melbourne, VIC, Australia
24. Minh TN, Wolters L (2010) Using historical data to predict application runtimes on backfilling parallel systems. In: Presented at the 18th Euromicro conference on Parallel, Distributed and Network-based Processing (PDP '10), Pisa, Italy
25. Duan R, Nadeem F, Wang J, Zhang Y, Prodan R, Fahringer T (2009) A hybrid intelligent method for performance modeling and prediction of workflow activities in grids. In: Presented at the 9th IEEE/ACM international symposium on Cluster Computing and the Grid (CCGRID '09), Shanghai, China
26. PANORAMA (2015) An approach to performance modeling and diagnosis of extreme scale workflows. Ewa Deelman, Christopher Carothers, Anirban Mandal, Brian Tierney, Jeffrey S. Vetter, Ilya Baldin, Claris Castillo, Gideon Juve, Dariusz Król, Vickie Lynch, Ben Mayer, Jeremy Meredith, Thomas Proffen, Paul Ruth, Rafael Ferreira da Silva. Int J High Perform Comput Appl (in press)

Chapter 8
Workflow Integration and Orchestration, Opportunities and the Challenges

Abstract Workflow orchestration is a method which smartly organizes the enterprise function with application, data, and infrastructure. The applications as well as their infrastructure can be dynamically scaled up or down using orchestration. On the contrary, integration enables the development of new applications with the capability to connect to any other application through specified interfaces. In this chapter, firstly, the opportunities and challenges in workflow orchestration and integration are explained. Following that, BioCloud, an architecture that demonstrates the task-based workflow orchestration using two bioinformatics workflows is explained in detail.

8.1 Introduction

Generally, the workflow systems can be either task-based or service-based. The task-based system mainly concentrates on mapping and execution of tasks, whereas the top-level orchestration is done by workflow engines. In contrast, the service-based system defines interfaces for a group of services, which is used to create new applications or to integrate with other application components. Orchestration and integration are two challenging processes of workflow execution in a cloud environment. Workflow orchestration is a process that intelligently lines up the business function among application, data, and infrastructure. With orchestration, an application as well as its infrastructure can be dynamically scaled up or down along with dynamic data integration. The orchestration process uses compose, combine, and connection approaches to deliver the tangible services. In contrast, integration process develops new applications for the users and provides the capability to link several components that are already deployed by satisfying their interfaces. This chapter mainly concentrates on several opportunities and challenges while executing workflows in a cloud environment.

Section 8.2 discusses about the steps involved in a workflow life cycle. Firstly, it discusses the workflow creation phase, wherein the details of workflow composition, representation, and data/control association are highlighted. Secondly, it emphasizes several approaches of workflow mapping. Following that, details about workflow execution, metadata, and provenance are explained.

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_8

Section 8.3 highlights various challenges and opportunities in workflow integration and orchestration. This section categorizes the challenges involved in executing collaborative scientific workflows into architectural, integration, computing, data management, and languages.

Section 8.4 explains the architecture of BioCloud, which supports task-based workflow execution that are developed using the Galaxy workflow system by leasing cloud instances from multiple cloud vendors. Additionally, how BioCloud overcomes most of challenges that are pointed out in Sect. 8.3 is also given in detail.

Finally, this chapter is closed off with several concluding remarks.

8.2 Workflow Life Cycle

Significant advancements in science and engineering usher the computation and data manipulation as its integral part of analysis. However, the computations may involve several steps, where each one requires different software and data sources. Further, the computations and their data may also be scattered in a distributed environment. These distributed computations pose several coordination and management challenges during the execution and data movement. Scientific workflows loomed as an archetype to coordinate and manage distributed computations and thus speed up the analysis. The workflow life cycle specifies the definition, execution, management, and reuse of workflows in its entirety. In general, the workflow life cycle has four different stages:

- Workflow creation
  – Create as well as represent the abstract workflow and associate data with abstract workflow to create executable workflow.
- Workflow mapping
  – Associate resources to abstract workflow.
- Workflow execution
  – Execute the mapped workflow over the associated resources.
- Metadata and provenance
  – Record the metadata and provenance information for reusing.

The overall flow of the workflow cycle is given as follows (refer to Fig. 8.1): The life cycle starts with workflow template, which clearly specifies the steps as well as the application components involved in analysis. These workflow templates can either be created from scratch by the users or obtained from workflow and component libraries. Further, these templates can be used as is or can be modified based on their demands. Moreover, this phase also facilitates collaboration, validation of overall logic, and its correctness of templates. However, these templates do not

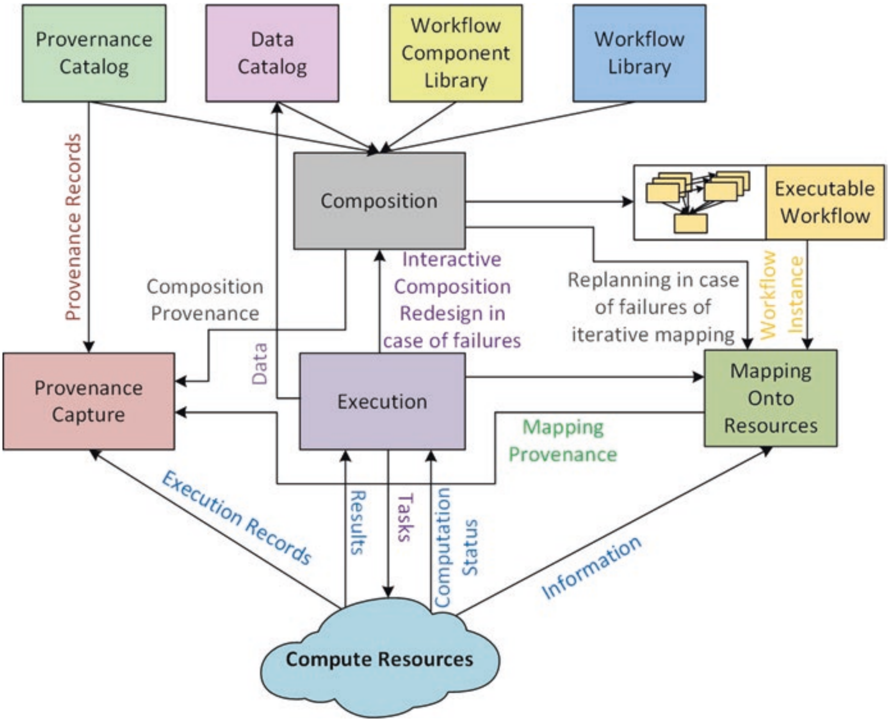


Fig. 8.1 Workflow life cycle

populate with their data at this stage. Next, the workflow instance is created by populating the data to the workflow templates using the data and metadata catalogs. At this stage, the workflow instances do not know the operational details of resources to carry out the analysis. Following that, the executable workflow is created by associating the workflow instance over the available distributed executing resources. The association step has to find out the appropriate software, hardware, and replicas of data mentioned in the workflow instance. Apart from this, the association phase is focused on the workflow restructuring to improvise the overall performance and workflow transformations to manage the data as well as provenance information. In concise, the executable workflow creation requires information about the available resources and data replicas on one side, whereas the application components requirements on the other side. After the successful creation of executable workflow, workflow engine started to execute the activities over the resources associated in the workflow. During execution, the data, metadata about data, and provenance information are collected and stored in a repository. This information enables reusing of workflows as is or adapt and modify it based on the future needs of users. Besides, the workflow cycle allows users to start from any stage. The following subsections explain each stage of workflow life cycle in detail.

8.2.1 Workflow Creation

The workflow creation involves composition, representation, and control or data association. The workflow composition allows the user to design or edit the overall analysis in an abstract or concrete way along with their dependencies. Here, the computational steps as well as their data input are specified in a single phase or two phases, i.e., firstly, create the workflow template and then instantiate with data. The workflow templates can be created from scratch or obtained from a shared repository that can be modified based on the need. Generally, the composition can be done in the following ways: textual, graphical, and semantic models.

The textual composition model uses a specific workflow language [1–4] to compose and edit workflows using a text editor. Conversely, composing the workflows in hand is tedious and error-prone, especially in scalable workflows like, parameter sweep application workflows. Alternatively, some workflow applications [4–8] provide a Web form, wherein users specify their attributes, which is used to automatically generate a workflow instance. The graphical composition model [9–13] provides an elegant graphical tool to reduce the developmental efforts of users to compose the workflow. Here, the nodes of a graph correspond to tasks, whereas edges stand for control or data dependencies. But, this model is not an alternative for textual model since it is tedious to graphically represent far complex scientific workflows which contain more than few dozens of tasks. Semantic composition model [14, 15] composes the semantic representations of workflow components with the formal workflow characteristics which are defined using domain ontologies and constraints. Additionally, it ensures that the workflow components, dataset, and data source are selected as per the restrictions given in workflow templates. The workflow representation clearly expresses the functional units and their order of execution based on their dependencies using various workflow representation languages [16–18]. Mostly, the workflows are represented using Directed Acyclic Graph (DAG), Petri nets, or Unified Modeling Language (UML). After successful composition and representing the workflows, control or data dependencies between the functional units need to be precisely expressed. To express control dependency, the control flow language should support branching (conditional and unconditional) and looping (simple and repetitive) to seamlessly transfer the control among functional units as in XBaya [19]. This feature precisely decides the branch to which the control has to be transferred as well as the duration of control to remain in the loop of that branch. However, the data dependency simply makes sure that all the data producer tasks are finished before the starting of data consumer.

8.2.2 Workflow Mapping

Workflow mapping is a procedure, which converts the abstract workflow into executable workflow. Alternatively, it modifies the resource-independent workflow instance into resource-dependent workflow. The appropriate services or resources for each functional unit can be either chosen by the user directly or using a scheduler. For service-based workflows, the scheduler finds and binds suitable services using their metadata, functional and nonfunctional properties by considering the Quality of Service (QoS) requirements of each functional units. In case of task-based workflows, the scheduler optimally chooses suitable resources, those satisfying the requirements of functional units. Here, the resources may be associated with the functional units either statically before starting the execution or dynamically at runtime. Further, the scheduler may use any of the following strategies to map the resources: just-in-time planning, full-ahead planning, or partial planning [20] to enable data reuse, reduced cost, and data usage. The just-in-time planning maps resources to each functional units thereby enhancing resource utilization and cost reduction. In full-ahead planning, the entire workflow is scheduled on the whole which enables the data reuse and thereby reduces the computation time. The entire workflow is sliced into multiple sub-workflows and map each sub-workflow to resources. However, the objective of optimization of the scheduler can be anything as specified in Chap. 3.

8.2.3 Workflow Execution

Finally, the resource-dependent workflows are starting to execute on the associated resources using a workflow engine [20]. During execution, the computations may carry out either on local resources or resources in a remote distributed environment. Further, the local or remote systems may be either an individual system or a cluster of resources managed by batch schedulers [21–23]. In either case, the workflow engine uses environment-specific adapters to transfer the executables and their dependencies such as inputting files to the executing site and monitoring the resource consumption and status of jobs. For instance, in the Globus toolkit [24], if a job is executed in an individual local machine, it uses a fork adapter, or if it is a cluster managed by Portable Batch System (PBS) [21], it uses a PBS adapter for the abovementioned steps. In contrast, if it is a multiple cloud environment, any multi-cloud library like Deltacloud [25] will be used for the creation and execution of cloud instances to execute the jobs. Besides, a service-based workflow execution is also supported in the Triana [26] workflow engine using a grid application prototype (GAP) interfaces to Web, WSRF [27], and P2P services. Similarly in cloud computing, some of the service-based workflow steps may be executed using Software as a Service (SaaS) and can be easily integrated with task-based steps that are executed using Infrastructure as a Service (IaaS). In the meanwhile, some of the workflow steps may be failed in a task-based workflow or a service do not complete its execution even after a timeout in a service-based workflow. Obviously, the execution model should be equipped with appropriate fault tolerance mechanisms. In task-based workflows, the fault tolerance is realized by saving the execution state (i.e., check pointing and migration) of tasks and migrate it to another resource (if needed), which is identified using rescheduling. Here, the check pointing can be done at three different levels: task level, application level, or operating system level (i.e., VM level). But, the service-based workflows reschedule the service after a timeout. Besides, the workflow engine has the capability to analyze the trace information or log files to identify the root cause of failure, if the task completions status is "failed."

8.2.4 Metadata and Provenance

The data provenance maintains the historic details about the newly created data object which provides the opportunity to reproduce the results. The provenance data contains the information about nodes that are clustered, input data chosen for execution, intermediate data chosen for reuse, scheduled execution sites, timestamp, software and library version, etc. The Karma [28] provenance system offers the capability to search over the provenance database that are collected using event-driven streams. However, most of the workflow systems optimize the user workflows by restructuring. Hence, it is tedious to directly use the provenance data over restructured workflows for reproducibility. With this objective, Pegasus is combined with PASOA provenance system [29].

With so many workflow creation, mapping, execution, metadata, and provenance methodologies, it is pretty hard to integrate and orchestrate scientific workflows. Mostly, the different parts of workflow applications require different resources: one step may need user intervention, the other step needs a data stream as input for its computation, and some steps need high-performance resources. None of the existing workflow systems can handle all sorts of demands on the whole. Therefore, the developers rely on multiple workflow engines to design and develop their workflows. Unfortunately, the workflows represented for one workflow system cannot be directly executed over the other because of the interoperability issues. At the bottom level, it is mandatory to provide the interoperability at the workflow description level. Consequently, the workflow described for one workflow system can be reused on the other without any modification. Section 8.3 discusses the opportunities and challenges in workflow integration, orchestration, and execution in cloud environment.

8.3 Challenges and Opportunities

Currently, most of the scientific research projects need collaboration from several domain experts who are usually available at different geographical locations. Alternatively, the part or complete scientific process needs collaboration among several research groups, datasets, and computational tasks. Hence, to execute a collaborative scientific workflow, the collaborators need to be interconnected through the Internet. However, storing, transferring, and reusing the intermediate data among the collaborators is a critical issue. Since cloud computing offers scalable data and computing resources which can also be shared among group members, these collaborative scientific workflows can be defined, developed, and executed using cloud resources. Nevertheless, the integration and orchestration of workflows pose the following challenges [30]: architectural, integration, computing, data management, and languages.

Architectural Challenges Scientific workflows are distributed in nature which opens new avenues for scalability in several dimensions like the number of users, use cases, and resources. Besides, each distributed part of the workflow may be described and executed in heterogeneous environments. In addition, these distributed parts need to be seamlessly integrated to complete the scientific process. Hence, the workflow management system (WfMS) which manages the workflow execution should architecturally support flexibility, extensibility, and portability. At the fabric level, it has to support heterogeneous services, software tools, data sources, and computing environments, which can be accessed in a distributed manner. Consequently, it has to manage task execution and their datasets along with their provenance data. Further, at the macro-level, it has to monitor the workflow execution to gather their resource consumption patterns, tasks execution status, and fault tolerance. Likewise, it has to provide interoperability among other WfMS at a workflow description level hereby offloading sub-workflows to execute over the fabric layer of other WfMS. Finally, it has to furnish the customization and interaction support at the user interface.

Integration Challenges The integration challenges focus toward the disputes evolved while executing the workflow over cloud computing resources. Here, the jobs in workflow may be either service-based or task-based. On the other hand, the services for the task units in a workflow from the cloud environment can be applications, services, software tools, compute, and storage resources. Therefore, it is mandatory to identify the target job type (i.e., service or task), and then it needs to be scheduled and dispatched to the appropriate cloud service for execution. Nonetheless, this step demands tweaking of WfMS architecture in several aspects: interface with several cloud providers for resource provisioning, workflow monitoring, seamless integration of sub-workflows to transfer or interconnect intermediate data, and job migration. Firstly, the jobs need computing resources or services that are available in cloud environment for their execution. But most of the current WfMS cannot directly interface with several cloud vendors to create the computing resources.

Likewise, choosing an appropriate cloud instance and the required number of such instances is also critical since it involves cost of execution. Apart from this, the WfMS needs interaction with cloud instances to monitor, debug, and collect the provenance data. For service-based jobs, the WfMS needs to identify suitable cloud services based on their metadata and Quality of Services (QoS). Briefly, the WfMS needs to be re-architected in such a way to interact with several cloud vendors to create resources at the first level along with the capability to interface with individual cloud computing instances or services at the next level to monitor and gather the resource information. BioCloud [31] is one such broker which reengineered the Galaxy WfMS [32] to integrate it with cloud vendors as well as cloud instances directly which is explained in detail later in Section.

Language Challenges Several parts of the workflow that are executed in a distributed manner can be integrated together using ad hoc scripts. These scripts can be of MapReduce, SwiftScript, and BPEL based. But, it has the capability to integrate the input data with a service or task, Further, it has to support scalability in terms of compute or storage resources to do the computations in parallel.

Computing Challenges As already mentioned, choosing an appropriate cloud instance type and the number of such instances are critical in workflow scheduling. After choosing the hardware, suitable machine images need to be selected for executing the workflow tasks is the next challenge. Then, the instances have to be configured dynamically with several parameters such as IP address, machine name, and cluster creation. Apart from this, the input data need to be staged into newly created instances before initiating the task execution. Further, different portions of the workflow demand different instance types and transfer of the intermediate data from one stage to other.

Data Management Challenges The primary bottleneck in data-intensive workflow execution is the mechanism to handle the data staging to and from the computing resources. Also, relative location of data and computational resources as well as their I/O speeds seriously affect the scalability of applications. Additionally, selecting an appropriate data source for the computation is also a challenging one. Mostly, the separate handling of data and computational-related issues leads to a huge amount of data staging between them. The collective management of data and computational resources along with their provenance data access patterns that consist of data locations, intermediate data, and the methodology to generate the data product are used to improve the scalability and performance.

Service Management Challenges Since service-oriented architecture (SOA) offers abstraction, decoupling, and interoperability among services, it can be easily leveraged in distributed scientific and engineering applications. However, it is hard to manage a huge number of services in terms of service invocation, state management, and service destruction. Similarly, data staging from one service to another is critical by considering performance and throughput.

8.4 BioCloud: A Resource Provisioning Framework for Bioinformatics Applications in Multi-cloud Environments

The significant advancement in next-generation sequencing (NGS) have enabled the generation of several gigabytes of raw data in a single sequencing run. This amount of raw data introduces new scalability challenges in processing, storing, and analyzing it, which cannot be solved using a single workstation, the only resource available for the majority of biological scientists, in a reasonable amount of time. These scalability challenges can be complemented by provisioning computational and storage resources using cloud computing in a cost-effective manner. BioCloud encapsulates all the complexity of resource management and provides a single entry point to create custom workflows and run them in a simple and efficient manner through its user-friendly Web interface. The public virtual machine (VM) image [33] can be employed to start a BioCloud instance. Here, BioCloud users have an existing account in at least one of the cloud providers. In order to start using BioCloud, users create a BioCloud account through the Web interface and complete their profiles by providing the available resources to be used. The resources can be cloud account(s), local clusters, servers, and datasets. Then a BioCloud instance is started on the cloud using the provided cloud credentials. Once the instance is initialized, the workflow manager interface (Galaxy [32]) is presented to the user which runs on one of the resources provided by the user. The workflows created by the user are executed over the computational resources defined earlier. If multiple computational resources are available, jobs in a multistep workflow can be run on different resources based on the scheduling algorithm and the user requirements.

BioCloud strives to exploit parallelism to reduce the overall workflow execution time by running parallel steps using different computing resources or dividing a single step into multiple parallel steps by partitioning the input data and computation, whenever possible. Also, it offers a loosely coupled architecture through its service-oriented architecture. BioCloud Portal Web service is employed to expose some of the functionalities of the system so that some of the workflow decisions (i.e., when to dispatch a workflow step and where to run this step) are delegated to the BioCloud Portal Web service. This enables modularity where scheduling logic is separated from the core workflow system. This provides the flexibility of updating the scheduling algorithm and other features of the system (i.e., improving abstract workflows submitted by the user and presenting the new workflow for execution) without requiring a software update on the user side.

8.4.1 BioCloud System Design

BioCloud follows a service-oriented architecture and consists of two main components, namely, BioCloud Portal and BioCloud Workflow Manager (BCWM). BioCloud Portal implements and encapsulates functions to orchestrate the workflow execution across disparate platforms. These functions are exposed as a service to enable interoperability and flexibility across platforms. BioCloud Portal also hosts a Web application to register and start using the system. BioCloud Portal can be regarded as the single access point of the BioCloud for all users (virtual organizations). BCWM, on the other hand, is the workflow management component of the system. While a unique BioCloud Portal is employed for all virtual organizations (VOs), an exclusive BCWM is created and employed for each VO. Although the virtual machines are provided for both components, it is also possible to use a custom workflow manager to consume BioCloud Portal services due to the loosely coupled architecture.

A VO is regarded as a single entity (organization) with multiple users. Once a VO is registered to BioCloud, the admin of the VO can manage its own users. It is noted that BioCloud does not provide computing resources, and it is assumed that VO has its own resources, which can be a cloud provider account, local cluster, or a personal computer. BioCloud uses the VO-provided resources to host the BCWM and run the workflows. A VO may have multiple users, sharing datasets and workflows. The overall BioCloud system architecture for multiple VOs is illustrated in Fig. 8.2. It is possible to start using the system through the Web application hosted on BioCloud Portal. Upon registration, VO can simply define the resources to be used by BioCloud. These resources will be employed to host BCWM for the corresponding VO and run the workflows. VO can select the resource to host the BCWM. If a cloud resource is selected, BioCloud initializes the BCWM instance using the preconfigured BCWM image in the corresponding cloud provider. Once the BCWM is initialized, BioCloud Portal enables the link to access the BCWM so that the VO can visit BCWM in a seamless manner without leaving the Web page. A local cluster or a PC can also be used to host the BCWM if OpenStack is available on the target system. A single unique BCWM instance is employed per VO to avoid data replication across multiple cloud providers and local clusters defined by the VO. This is realized without sacrificing the ability of using resources in multiple cloud providers simultaneously which is the key contribution of the presented system. Based on the workflow and the available resources, BioCloud can determine the computational resources to be used for particular steps.

Web Interface and User Interaction Despite the underlying distributed architecture of multiple components (i.e., BioCloud Portal and BCWM), users access BioCloud through a single, unified Web interface. This user-friendly Web interface enables users to manage resources (i.e., local clusters and cloud services), design and run workflows, and collect results. Workflow management component of BioCloud is extended from Galaxy [32] so that multistep pipelines can be created in a simplified manner. It is possible to specify the computational resource to be used

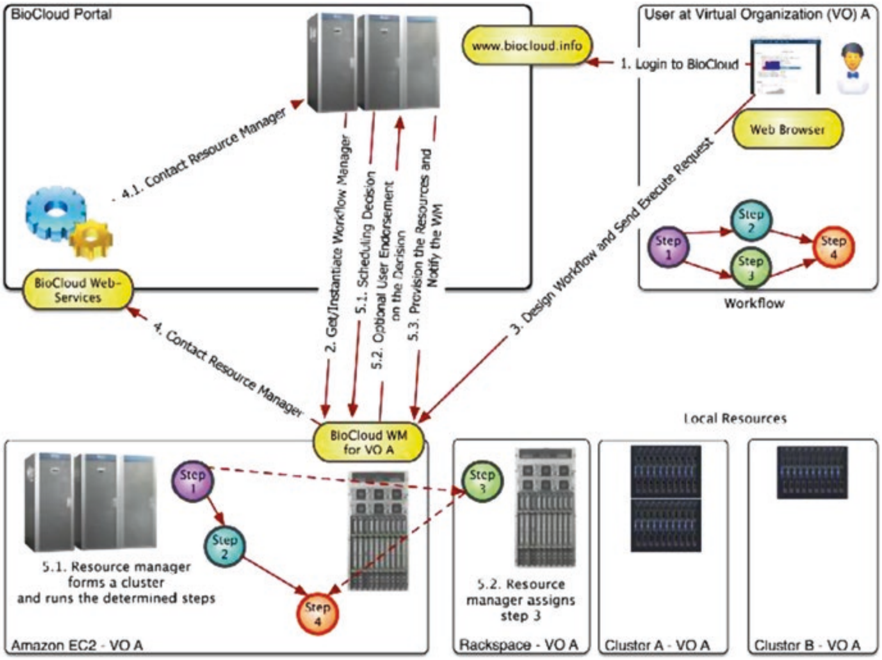

BioCloud Portal
www.biocloud.info
User at Virtual Organization (VO) A
1. Login to BioCloud
Web Browser
4.1. Contact Resource Manager
2. Get/Instantiate Workflow Manager
5.1. Scheduling Decision
5.2. Optional User Endorsement on the Decision
5.3. Provision the Resources and Notify the WM
3. Design Workflow and Send Execute Request
Step 1
Step 2
Step 3
Step 4
Workflow
BioCloud Web-Services
4. Contact Resource Manager
BioCloud WM for VO A
Local Resources
Step 1
Step 2
Step 4
Step 3
5.1. Resource manager forms a cluster and runs the determined steps
5.2. Resource manager assigns step 3
Amazon EC2 - VO A
Rackspace - VO A
Cluster A - VO A
Cluster B - VO A

Fig. 8.2 BioCloud system design

for a particular workflow step. Unless a particular resource is specified, BioCloud exploits its scheduling algorithm to designate the resources to be used for each step based on the workflow and user requirements such as cost and time. The details are provided later in this chapter.

Workflow Management and Execution BioCloud workflow management component (BCWM) enables workflow creation in a drag-and-drop fashion thanks to the underlying Galaxy [32] application. On the design pane, each workflow step is represented by an independent box. Each box is associated with a tool (i.e., application) to be used in the corresponding workflow step. Two boxes can be connected with a directed edge to indicate the dataflow and the resulting data dependency between them. Incoming and outgoing edges connected to the box represent input and output data, respectively. Data dependency between the workflow steps require preceding steps to be completed before initiating the following steps. Galaxy assumes execution of the whole workflow on a single resource which also hosts Galaxy and cannot ensure data dependency when multiple resources of various cloud providers and local clusters are to be used. BioCloud eliminates such constraints and enables running different workflow steps on different resources simultaneously or sequentially. BioCloud segregates workflow management and workflow execution through its service-oriented architecture. While Web services in BioCloud Portal is responsible for determining the resources to be used for each workflow step and ensuring availability of the resources through resource management and provisioning, BCWM is responsible from dispatching workflow steps for execution on the resources predetermined by BioCloud Portal and monitoring them.

Initially, BCWM informs BioCloud Portal about the workflow to be run by sending the related information through the Web services. In order to designate the workflow execution schedule and determine the resources to be used at each step, BioCloud scheduler is employed by BioCloud Portal. Scheduler receives the submitted workflow as a DAG (Directed Acyclic Graph) along with the associated tool names at each step and input data size. One of the key features of the scheduler is inherent workflow improvement through data partitioning and parallelism. The scheduler automatically manipulates the DAG to enable parallelism. It employs a profiler to estimate expected running times of the tools, the amount of data to be produced, and the cost of execution to be incurred considering available resources. Based on this information, the scheduler identifies the resources to be used at each step considering cost and time requirements. It employs a resource manager to ensure availability of the resources before executing a workflow step. When the resources are provisioned, BCWM is allowed to dispatch the next available workflow step for execution. This enables dynamic scaling up of the resources right before the execution of a particular workflow step to meet the resource demand of the corresponding VO. The resource management module also tracks the provisioned resources to scale down based on the supply-demand balance in the next billing cycle of the provisioned resources.

BioCloud not only provides an efficient scheduler to minimize the execution cost while meeting cost and time requirements which is the key contribution of this paper, it also offers a user-friendly platform to encapsulate the complexity of identifying resources to be used among several options, using resources simultaneously on multiple cloud providers to execute workflows while handling data partitioning and parallelism, dynamic resource scaling, and cluster configuration in the cloud. Hiding such a complexity from the user enables her to focus on the workflow design. The user simply clicks the run button to execute the workflow. Figure 8.2 depicts the steps involved on execution of a workflow where the user has Amazon EC2 and Rackspace cloud accounts as well as local clusters.

Authentication Alberich policy engine [34] is leveraged by BioCloud to authenticate the users based on the defined roles and permissions. BioCloud Portal exploits permissions of the account provided by the user for the initial deployment of the BCWM and to run the workflows. Considering the fact that various steps of the workflow can be executed using different computational resources, the Alberich policy engine authenticates a user for the particular resource. The users provide cloud service credentials so that the Alberich policy engine retrieves roles, permissions, and privileges to authenticate and authorize users for the resource pools. Accessing the image details, profiler information, resource information, and the allowed actions are determined based on the access rights. The policy engine is extended so that the resources from different cloud resources can be used.

Profiler Scheduling distributed applications can be challenging in a multi-cloud environment due to the lack of knowledge about the application characteristics. In order to realize a versatile multi-cloud scheduling algorithm, the knowledge about the application's runtime behavior on various resources is needed. Besides, not all the applications exhibit same kind of resource consumption pattern in all stages. Thus, looking into the resource consumption pattern, extracting the knowledge and classifying the applications can assist the scheduling algorithm for a better decision-making in a multi-cloud environment. In BioCloud, a profiler component monitors the resource consumption of applications and stores it in a profile database. BioCloud monitors execution of the workflow steps individually and collects profiling information such as running time and output file size for the tool used in the corresponding workflow step considering the resources exploited such as CPU, memory, number of compute nodes if a cluster is used, input file size, etc.

Resource Manager Resource manager is a component to collect the resource information about various resources hosted in multiple cloud environments periodically. Resource information includes, but is not limited to, hardware, OS image, network, secondary storage, and memory of all the instances present in multiple clouds. For advanced metrics, the instances are enabled with cloud monitoring tools such as cloud watch and then these metrics are up-streamed to the broker via Deltacloud APIs. This resource information together with image information gives the unified view about the multi-cloud environment that will be used by the scheduler.

Image Manager Image manager is a component to collect the available VM image information from multiple cloud providers. This information includes but is not limited to operating system, metadata about the software installed, and the description of all the images present in multiple clouds. It can be collected from the images deployed in various cloud providers using Deltacloud API via controller. BCWM uses image manager to launch new instances.

Scheduler Dynamic nature of the multi-cloud environment and availability of a wide variety of resources with diverse characteristics and capabilities demand provisioning appropriate set of resources in a dynamic manner in order to satisfy the requirements of an application. Some of the recent works focus on managing applications modeled as bag of tasks. For example, the scheduling algorithm which uses a linear programming model to calculate the optimal deployment configuration. The scheduler adds and eliminates instances based on the incoming requests. Besides, some of the works discussed in Sect. 8.4 focus on the bag of distributed tasks and introduce a heuristic algorithm that takes the location of the running tasks and their data sources into account. In contrast to these studies, the resource provisioning scheduler in BioCloud manages the workflow as a Directed Acyclic Graph (DAG).

Efficient cloud computing requires solving multi-objective combinatorial problems such as partitioning and scheduling. Here, the goal is to provide a scheduler which considers cost and time to complete tasks so that the resources are allocated in such a way that the execution time is reduced for the given budget, while the throughput and resource utilization is improved. Therefore, the scheduler should be aware of the cost model, resource availability, and favorable submission time of all the cloud providers to estimate the cost involved in resource schedule. It should have the capability to estimate the completion time of an application using profiling information of the tools based on the earlier executions. One can model the execution time and utilize that for deciding the optimum number of resources under different scheduling scenarios.

The BioCloud scheduler regards the submitted workflows as DAG and aims to maximize parallelization. For steps that can be executed in a data-parallel manner, BioCloud partitions the data and hence the associated computation as much as possible to decrease the overall running time. Once the user submits the workflow for execution, BioCloud partitions the data to match with the available resources. The details of the scheduling algorithm is provided in the next section. Galaxy [32] also provides a partitioning capability for the tools. However, it cannot fully optimize the execution of workflows where two (or more) consecutive steps can be run using the partitioned data. In such cases, Galaxy would redundantly merge and partition the data in between the consecutive steps. BioCloud scheduler aims to designate a schedule for the given abstract workflow so that the execution of the improved workflow can be completed within the given deadline using the supplied budget. The scheduler also determines the resources to be used in each workflow step and cooperates with the resource manager to ensure provisioning of these resources.

8.4.2 BioCloud in Action

Here, two different scenarios are presented to demonstrate the smooth transition from a single workstation, the only resource available for the majority of biological scientists, to a multi-cloud environment. In the first scenario, only a single workstation is assumed to be available to the user. On the other hand, besides the workstation, multiple cloud resources are also available in the second scenario. Two different cloud vendors are selected to demonstrate the flexibility and interoperability of BioCloud. These scenarios are tested with two different use cases (bioinformatics workflows) as explained below.

The testbed consists of a local workstation and multiple instances from two cloud vendors: Amazon and Rackspace. The workstation is equipped with two Intel Xeon E5520 CPU clocked at 2.27 Hz and 48 GB memory. Each CPU is a quad-core, with Hyper-Threading enabled and all cores share an 8 MB L3 cache. The "instance-type" BioCloud used on Amazon EC2 is M3 General Purpose Double Extra Large (m3.2xlarge). This configuration has 8 cores with a memory of 30 GB. The "flavor" selected on Rackspace is Performance 2 with a memory of 30 GB and 8 cores. More information regarding the configuration of these particular instance types can be found in the respective cloud vendors' websites. BioCloud forms a dynamic cluster, on the fly, upon needed in the corresponding cloud service using instances of these types.

ExomeSeq Workflow The first use case of BioCloud is an exome sequence analysis pipeline obtained from their collaborators [35], which is known as ExomeSeq. Test and control data with paired-end reads are used as input. In the first two steps, sequencing reads are aligned to human reference genome using BWA [36] alignment and BWA sampe steps. The third step sorts the alignments by leftmost coordinates. Duplicates are marked in the following two steps using two different tools. In step six, local realignments around indels are performed and the last step detects indels. The abstract ExomeSeq workflow is depicted in Fig. 8.3. The readers can refer to [35] for more information regarding the workflow steps. As mentioned earlier, one of the key features of BioCloud is its inherent workflow improvement facility through data partitioning and parallelism. BioCloud scheduler evaluates the submitted abstract workflow and generates an optimized workflow that would utilize available resources and hence enable parallelism. For example, for the abstract ExomeSeq workflow designed by the user (Fig. 8.3), BioCloud designates an improved version of this workflow as given in Fig. 8.4. Here, BioCloud scheduler checks whether data partitioning can be enabled for workflow steps, and considering available resources and profiling data of earlier executions, BioCloud scheduler determines to dispatch the test data and the control data to separate cloud resources for execution. In this scenario, the workflow steps for test data are run in Amazon EC2, and the steps for the control data are run in Rackspace. The output data of step six are transferred back to the workstation, and the last step is executed locally where the final result will also be stored.

Transcriptome Assembly and Functional Annotation De novo transcriptome assembly and functional annotation are considered as an essential but computationally intensive step in bioinformatics. The objective of the assembly and the annotation workflow is to assemble a big dataset of short reads and extract the functional annotation for the assembled contigs. As can be seen in Fig. 8.5, the workflow consists of four stages. The first stage is data cleaning in which a Trimmomatic [37] tool is applied on the paired-end reads dataset. After that, the output is converted to FASTA format. In stage two, the assembly, the clean dataset is used as input to five different de novo transcriptome assemblers: Tran-Abyss [38], SOAPdenovo-Trans [39], IDBA-Tran [40], Trinity [41], and Velvet-Oases [42]. The assembled contigs from each assembler are merged and used as input to stage three which includes clustering and removing redundant contigs as well as applying reassembly for the unique contigs. In stage three, the TGICL tool [43] which utilizes MEGABLAST [44] is used for contigs clustering and CAP3 [45] for assembly. Functional annotation is done in the last stage, and it is the most computational part. The blast comparison and functional annotation used in this workflow follow the pipeline detailed in [46]. Three major sequences databases, NCBI nonredundant protein sequences (nr), UniProt/Swiss-Prot, and UniProt/TrEMBL, are used in the sequence comparison steps. The Blastx results are parsed in the last step, and their associated GO categories are generated. The used dataset is rice transcriptome data from Oryza sativa 9311 (http://www.ncbi.nlm.nih.gov/sra/SRX017631). 9.8M paired-end reads of 75 bp length and totaling 1.47 Gbp were generated using the Illumina GA

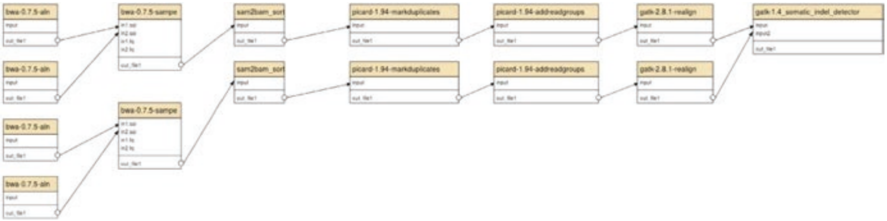


Fig. 8.3 Abstract ExomeSeq workflow

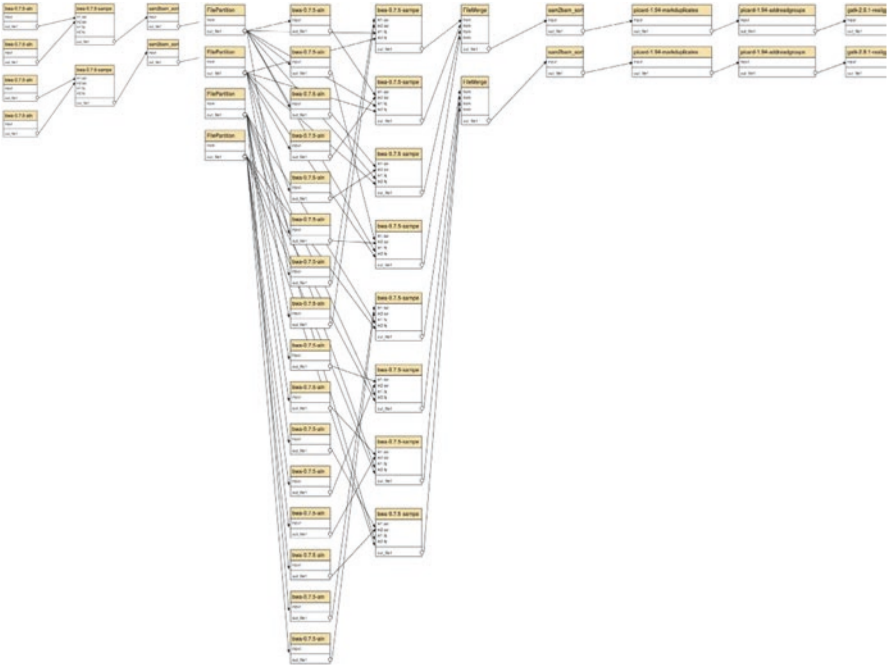


Fig. 8.4 Generated workflow

platform [47]. The output contigs of the TGICL step were filtered by removing contigs of length less than 400 base pairs. For practical issues, the number of sequences of the three protein databases was reduced to 1% of its original sequences count, and the databases were installed in the single workstation and remote clusters. Similar to ExomeSeq, transcriptome assembly and annotation use case are tested in two different scenarios. In the first one, the assumption is that the user has access to a single commodity workstation to run the workflow. The second scenario assumes the availability of multiple cloud resources besides the workstation.

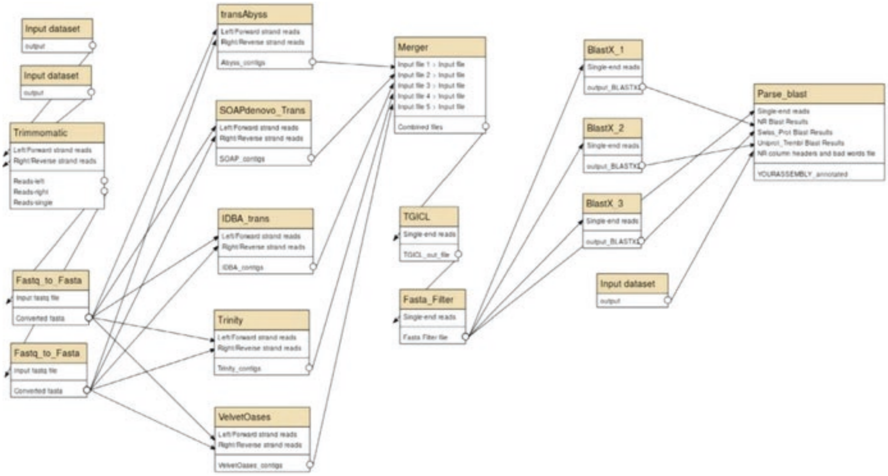


Fig. 8.5 Assembly and annotation workflow

8.5 Conclusion

This chapter initially spotlights two different classes of workflows together with their characteristics: task-based and service-based. Further, it explains several phases in the workflow life cycle. After that, several architectural, integration, computing, data management, and language-related challenges are elucidated in detail. Finally, the BioCloud which is built on top of the Galaxy workflow system to execute their task-based workflow by leasing cloud instances from multiple cloud vendors is described in detail. Chapter 9 expounds the automated scheduling of workflows through workload consolidation.

References

1. Andrews T, Curbera F, Dholakia H, Goland Y, Klein J, Leymann F, Liu K, Roller D, Smith D, Thatte S, Trickovic I, Weerawarana S, Business process execution language for web services version 1.1
2. Oinn T, Addis M, Ferris J, Marvin D, Senger M, Greenwood M, Carver T, Glover K, Pocock MR, Wipat A, Li P (2004) Taverna: a tool for the composition and enactment of bioinformatics workflows. Bioinformatics 20(17):3045–3054
3. Condor team, DAGMan: a Directed Acyclic Graph Manager, July 2005. http://www.cs.wisc.edu/condor/dagman/
4. Deelman E, Singh G, Su M-H, Blythe J, Gil Y, Kesselman C, Mehta G, Vahi K, Berriman GB, Good J, Laity A, Jacob JC, Katz D (2005) Pegasus: a framework for mapping complex scientific workflows onto distributed systems. Sci Program J 13(3):219–237

5. Berriman G, Good J, Laity A, Bergou A, Jacob J, Katz D, Deelman E, Kesselman C, Singh G, Su M et al, Montage: a grid enabled image mosaic service for the national virtual observatory. In: Astronomical data analysis software and systems, ADASS, XIII
6. Berriman G, Deelman E, Good J, Jacob J, Katz D, Kesselman C, Laity A, Prince T, Singh G, Su M (2004) Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand. In: Proceedings of SPIE 5493, pp 221–232
7. Lathers A, Su M, Kulungowski A, Lin A, Mehta G, Peltier S, Deelman E, Ellisman M, Enabling parallel scientific applications with workflow tools. In: Proceedings of Challenges of Large Applications in Distributed Environments, CLADE
8. Muench J et al, SCEC earthworks science gateway: widening SCEC community access to the TeraGrid. In: TeraGrid 2006 conference
9. Lord H (1995) Improving the application development process with modular visualization environments. ACM SIGGRAPH Comput Graph 29(2):10–12
10. Parker SG, Miller M, Hansen CD, Johnson CR (1998) An integrated problem solving environment: the SCIRun computational steering system. In: Proceedings of the 31st Hawaii International Conference on System Sciences, HICSS-31, pp 147–156
11. Altintas I, Berkley C, Jaeger E, Jones M, Ludäscher B, Mock S (2004) Kepler: an extensible system for design and execution of scientific workflows. In: 16th international conference on Scientific and Statistical Database Management, SSDBM. IEEE Computer Society, New York, pp 423–424
12. Taylor I, Shields M, Wang I, Harrison A (2005) Visual grid workflow in triana. Journal of Grid Computing 3(34):153–169
13. Callahan S, Freire J, Santos E, Scheidegger C, Silva C, Vo H (2006) Managing the evolution of dataflows with vis Trails. In: IEEE workshop on workflow and data flow for scientific applications, SciFlow
14. Maechling P, Deelman E, Zhao L, Graves R, Mehta G, Gupta N, Mehringer J, Kesselman C, Callaghan S, Okaya D, Francoeur H, Gupta V, Cui Y, Vahia K, Jordan T, Field E (2007) Workflows for e-Science. Springer, New York, pp 143–166. Ch. SCEC CyberShake workflows – automating probabilistic seismic hazard analysis calculations
15. Knight K, Marcu D (2005) Machine translation in the Year 2004. In: Proceedings of the 2005 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP, vol 5. IEEE Computer Society, New York, pp 965–968
16. ISO/IEC 15909-1, High-level Petri nets – Part 1: concepts, definitions and graphical notation, 2004
17. Fowler M, Scott K (1997) UML distilled. Addison-Wesley, Reading
18. Fletcher T, Ltd C, Furniss P, Green A, Haugen R, BPEL and business transaction management: choreology submission to OASIS WS-BPEL Technical Committee, published on web
19. Shirasuna S, XBaya workflow composer. http://www.extreme.indiana.edu/xgws/xbaya
20. Deelman E, Singh G, Su M-H, Blythe J, Gil Y, Kesselman C, Mehta G, Vahi K, Berriman GB, Good J, Laity A, Jacob JC, Katz D (2005) Pegasus: a framework for mapping complex scientific workflows onto distributed systems. Scientific Programming Journal 13(3):219–237
21. Henderson R, Tweten D, Portable batch system: external reference specification, Ames Research Center, Moffett Field
22. Zhou S (1992) LSF: load sharing in large-scale heterogeneous distributed systems. In: Proceedings Workshop on Cluster Computing, pp 1995–1996
23. Litzkow M, Livny M, Mutka M (1988) Condor – a hunter of idle workstations. In: Proceedings of the 8th international conference on distributed computing systems. IEEE Computer Society, New York, pp 104–111
24. The Globus Alliance. See web site at: http://www.globus.org
25. Deltacloud: https://deltacloud.apache.org/
26. Taylor I, Shields M, Wang I, Harrison A (2005) Visual grid workflow in triana. J Grid Comput 3(3–4):153–169
27. Czajkowski K, DF Ferguson, Foster I, Frey J, Graham S, Sedukhin I, Snelling D, Tuecke S, Vambenepe W (2004) The WS-resource framework, Technical Report, The Globus Alliance

28. Simmhan YL, Plale B, Gannon D (2006) Performance evaluation of the karma provenance framework for scientific workflows, in: International Provenance and Annotation Workshop, IPAW. Springer, Berlin
29. Miles S, Groth P, Deelman E, Vahi K, Mehta G, Moreau L (2008) Provenance: the bridge between experiments and data. Comput Sci Eng 10(3):38–46
30. Zhao Y, Fei X, Raicu I, Lu S (2011) Opportunities and challenges in running scientific workflows on the cloud. International conference on cyber-enabled distributed computing and knowledge discovery, Beijing, pp 455–462
31. Senturk IF, Balakrishnan P, Abu-Doleh A, Kaya K, Qutaibah M, Ümit V (2016) A resource provisioning framework for bioinformatics applications in multi-cloud environments. Future generation computer systems, Elsevier, (Accepted to Publish impact factor-2.64): doi:10.1016/j.future.2016.06.008
32. Goecks J, Nekrutenko A, Taylor J, Team TG (2010) Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. Genome Biol 11(8) http://dx.doi.org/10.1186/gb-2010-11-8-r86, R86+
33. BioCloud. URL http://confluence.qu.edu.qa/display/KINDI/BioCloud
34. Alberich. URL https://github.com/aeolus-incubator/alberich
35. Woyach JA, Furman RR, Liu T-M, Ozer HG, Zapatka M, Ruppert AS, Xue L, Li DH-H, Steggerda SM, Versele M, Dave SS, Zhang J, Yilmaz AS, Jaglowski SM, Blum KA, Lozanski A, Lozanski G, James DF, Barrientos JC, Lichter P, Stilgenbauer S, Buggy JJ, Chang BY, Johnson AJ, Byrd JC (2014) Resistance mechanisms for the bruton's tyrosine kinase inhibitor ibrutinib. New Engl J Med 370(24):2286–2294. http://dx.doi.org/10.1056/NEJMoa1400029, pMID: 24869598
36. Li H, Durbin R (2009) Fast and accurate short read alignment with burrows–wheeler transform. Bioinformatics 25(14):1754–1760
37. Bolger AM, Lohse M, Usadel B, Trimmomatic: a flexible trimmer for illumina sequence data, Bioinformatics. http://dx.doi.org/10.1093/bioinformatics/btu170
38. Robertson G, Schein J, Chiu R, Corbett R, Field M, Jackman SD, Mungall K, Lee S, Okada HM, Qian JQ, Griffith M, Raymond A, Thiessen N, Cezard T, Butterfield YS, Newsome R, Chan SK, She R, Varhol R, Kamoh B, Prabhu A-L, Tam A, Zhao Y, Moore RA, Hirst M, Marra MA, Jones SJM, Hoodless PA, Birol I (2010) De-novo assembly and analysis of RNA-seq data. Nat Methods 7(11):912
39. Xie Y, Wu G, Tang J, Luo R, Patterson J, Liu S, Huang W, He G, Gu S, Li S, Zhou X, Lam T-W, Li Y, Xu X, Wong GK-S, Wang J, SOAPdenovo-Trans: De novo transcriptome assembly with short RNA-Seq reads. Bioinformatics. http://dx.doi.org/10.1093/bioinformatics/btu077
40. Peng Y, Leung HCM, Yiu S-M, Lv M-J, Zhu X-G, Chin FYL (2013) IDBAtran: a more robust de novo de bruijn graph assembler for transcriptomes with uneven expression levels. Bioinformatics 29(13):i326–i334. http://dx.doi.org/10.1093/bioinformatics/btt219
41. Grabherr MG, Haas BJ, Yassour M, Levin JZ, Thompson DA, Amit I, Adiconis X, Fan L, Raychowdhury R, Zeng Q, Chen Z, Mauceli E, Hacohen N, Gnirke A, Rhind N, di Palma F, Birren BW, Nusbaum C, Lindblad-Toh K, Friedman N, Regev A (2011) Full-length transcriptome assembly from RNA-Seq data without a reference genome. Nat Biotechnol 29(7):652
42. Schulz MH, Zerbino DR, Vingron M, Birney E, Oases: Robust de novo rnaseq assembly across the dynamic range of expression levels, Bioinformatics. http://dx.doi.org/10.1093/bioinformatics/bts094
43. Pertea G, Huang X, Liang F, Antonescu V, Sultana R, Karamycheva S, Lee Y, White J, Cheung F, Parvizi B, Tsai J, Quackenbush J (2003) TIGR gene indices clustering tools (TGICL): a software system for fast clustering of large EST datasets. Bioinformatics 19(5):651–652. http://dx.doi.org/10.1093/bioinformatics/btg034
44. Zheng Zhang LW, Scott S, Miller W (2000) A greedy algorithm for aligning DNA sequences. Comput Biol 7:203–214. http://dx.doi.org/10.1089/10665270050081478
45. Huang X, Madan A (1999) Cap3: a DNA sequence assembly program. Genome Res 9:868–877. http://dx.doi.org/10.1089/10665270050081478

46. De Wit P, Pespeni MH, Ladner JT, Barshis DJ, Seneca F, Jaris H, Therkildsen NO, Morikawa M, Palumbi SR (2012) The simple fool's guide to population genomics via RNA-Seq: An introduction to high-throughput sequencing data analysis. Mol Ecol Res 12(6):1058–1067. http://dx.doi.org/10.1111/1755-0998.12003
47. Deelman E, Gannon D, Shields M, Taylor I (2009) Workflows and e-science: An overview of workflow system features and capabilities. Futur Gener Comput Syst 25(5):528–540

Chapter 9
Workload Consolidation Through Automated Workload Scheduling

Abstract Workload consolidation is an approach to enhance the server utilization by grouping the VMs that are executing workflow tasks over multiple servers based on their server utilization. The primary objective is to optimally allocate the number of servers for executing the workflows which in turn minimize the cost and energy of data centers. This chapter consolidates the cost- and energy-aware workload consolidation approaches along with the tools and methodologies used in modern cloud data centers.

9.1 Introduction

Workflow scheduling is a major problem within the heterogeneous cloud computing environment. The applications are represented as Directed Acyclic Graph (DAG) form, where each node in the DAG represents the computation and the edges will indicate the data transfer between tasks. After scheduling, these tasks are getting executed over several cloud instances that are hosted by different physical servers in a data center. This leads to the rebirth of underutilization problem in server hardware if each of them hosts only very few cloud instances (i.e., VMs). However, this problem can be easily sorted out by consolidating the VMs over minimum number of physical servers without compromising the QoS and shutdown rest of the servers. Hence, the primary objective of workload (i.e., VM) scheduling is to reduce the cost of execution by leasing minimum number of cloud instances and also to reduce the energy consumption through workload (i.e., VM) consolidation in data centers. With this focus, this chapter initially provides a comprehensive survey on cost- and energy-aware scheduling algorithms. Later, it provides the approaches that are practiced in modern cloud data centers to save energy and cost.

Section 9.2 highlights the elements of workflow reference model. Sections 9.3 and 9.4 provide a comprehensive survey of workflow scheduling with the focus of cost and energy, respectively.

Section 9.5 illustrates the detailed information, tools, and approaches used in modern cloud data centers to minimize the cost as well as energy consumption using VM consolidation.

Finally, Sect. 9.6 consolidates the significance and benefits of cost- and energy-aware scheduling.

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_9

9.2 Workflow Scheduling

Workflow scheduling is an assignment of mapping the workflow tasks on suitable resources while satisfying the constraints imposed by the users [1]. In other words, it is the process of automating the workflow execution with the help of algorithms. A workflow will consist of sequence of connected instructions. The major objective of workflow scheduling is to automate the events that are engaged in the process of sending the data and fields between the participants of the cloud maintaining the constraints [2]. The performance of the entire system can be achieved by properly scheduling the workflows with the help of the scheduling algorithms.

The Workflow Management Coalition (WfMC) defined workflow as "The automation of a business process, in whole a set of procedural rules" [3].The components of the workflow reference model are represented in Fig. 9.1.

Workflow engine: The workflow engine will provide a runtime environment to create, manage, and execute the workflow instances.

Process definition: The processes are defined in such a way that it facilitates the automated manipulation.

Workflow interoperability: Interoperability is provided between the different kinds of workflow systems.

Invoked application: It helps in the communication between the different kinds of IT applications.

Workflow client application: It supports for the interaction with the users with the help of user interface.

Administration and monitoring: It helps in coordinating the composite workflow application environment.

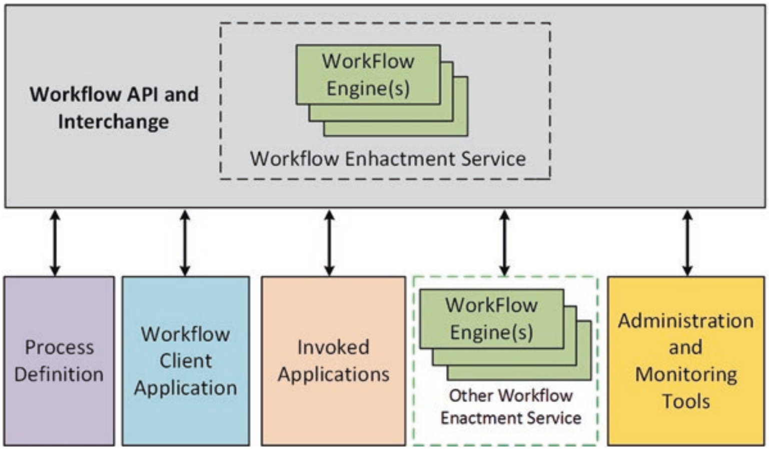

Workflow API and Interchange
WorkFlow Engine(s)
Workflow Enhactment Service
Process Definition
Workflow Client Application
Invoked Applications
WorkFlow Engine(s)
Other Workflow Enactment Service
Administration and Monitoring Tools

Fig. 9.1 Workflow reference model

9.3 Cost-Based Scheduling Algorithm

The two important workflow scheduling objectives considered in this section are cost and energy. The algorithm design and its comparison with other existing algorithms are also highlighted in the following sections.

9.3.1 Proportional Deadline Constrained Algorithm

The Proportional Deadline Constrained (PDC) algorithm helps to achieve the deadline constraints with the minimum cost. PDC will consist of the following four different steps:

Workflow leveling: Each task in the workflow is grouped into different kinds of levels.

Deadline distribution: The user-defined deadlines are divided and distributed in such a way that each level will get its own deadlines.

Task selection: A task is selected for execution based on its priority in the ready queue.

Instance selection: The instances are selected in such a way that the deadlines are met with the minimum cost.

In order to evaluate the performance of the PDC algorithm, the following scientific workflow structures are used:

1. CyberShake
2. Montage
3. LIGO

CyberShake: This seismological workflow was used by the Southern California Earthquake Center (SCEC) to characterize the earthquake hazards [4, 5]. The structure is shown in Fig. 9.2.

LIGO: Laser Interferometer Gravitational-Wave Observatory (LIGO) (refer to Fig. 9.3) is a data-intensive workflow which is used to find out the gravitational wave signatures in the data. Thus, the four stages in the PDC algorithm focus on the deadline constraints with minimum cost. With PDC viable schedule can be constructed even with tight deadlines and also with minimum cost. With the help of PDC algorithm, all the e-Science workflow scheduling like cumulus [6, 7] and nimbus has met their deadline with minimum cost. Overall PDC algorithm helps in achieving the deadlines with the lower cost.

Montage: Montage is an astronomy-based application in which the mosaic images are generated for the sky, thereby providing a detailed understanding of the portion of the sky. The structure is shown in Fig. 9.4.

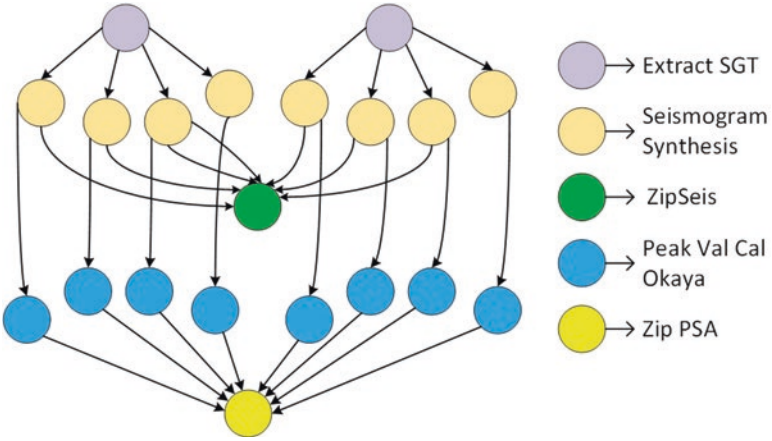


Fig. 9.2 CyberShake

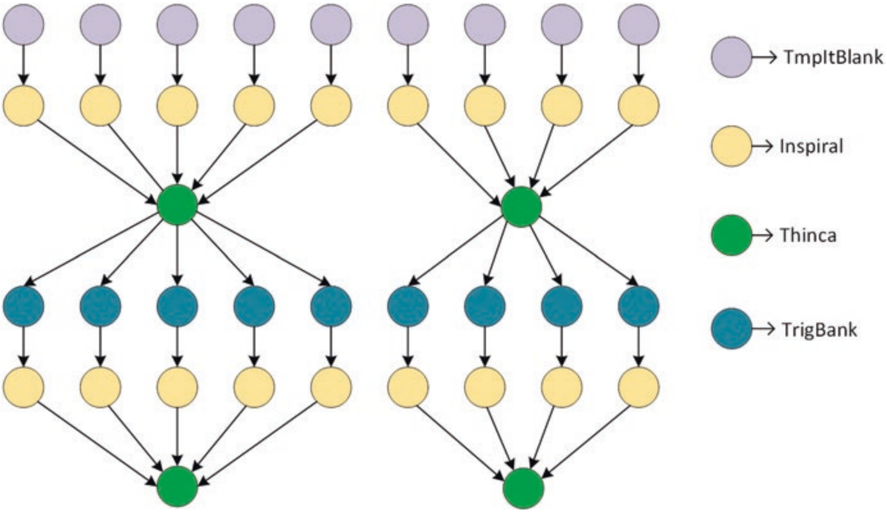


Fig. 9.3 LIGO

9.3.2 Particle Swam Optimization (PSO) Algorithm

PSO is a heuristic algorithm which will schedule the cloud workflow applications with minimum computational cost and data transmission cost. In PSO each particle will have their own fitness value, and it will be evaluated by the fitness function to get an optimized result.

PSO is a self-adaptive global search population-based algorithm without any direct recombination [8]. The social behavior of the particles is used in this algo-

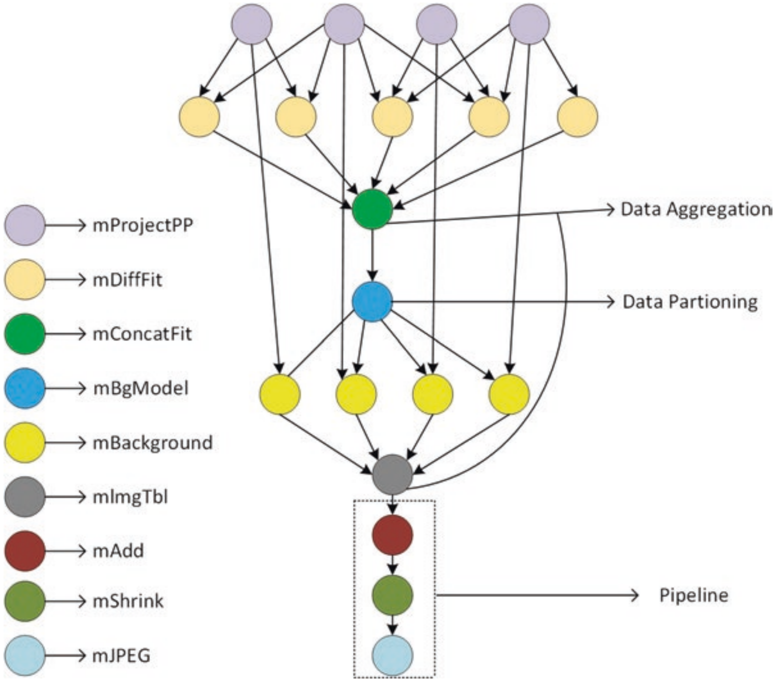


Fig. 9.4 Montage

rithm. PSO will generate a number of candidate solutions known as particles, and these particles will be moved around in a finite space. Initially the particles will move in its local best-known position and then in the global best of the entire population. Later these movements are found by the other particles also and it is termed as Swarm toward the best solutions.

PSO is started with a set of random particles, and then search for the optima is made by updating the generations. In every iteration each and every particle is updated with two best values. The first value is the fitness value achieved so far and it is termed as pbest. The second value is the best value of any particle within the flock and it is termed as gbest. After finding the pbest and gbest, the particle velocity (v) is adjusted with the help of the following equation:

$$v = v + c1^{*}\,\text{rand}^{*}\left(\text{pbest} - p\right) + c2^{*}\,\text{rand}^{*}\left(\text{gbest} - p\right) \tag{9.1}$$

where:

p = particle position
v = particle velocity
$c1$ = weight of the local information
$c2$ = weight of the global information
pbest = best position of the particles

gbest = best position of the swarm
rand = random variable

PSO Algorithm in Workflow Scheduling
The fitness function will evaluate each and every particle. Random initialization of the position and velocity is done for the particles. Then the fitness value is calculated using the fitness functions, and the local best position will be evaluated which is then followed by the updation. After computing the mappings using the PSO, the scheduling algorithm will dispatch the ready tasks to their corresponding resources. A ready task is a task which has completely received all its data from its parent tasks. With the help of PSO algorithm, both the cost and the makespan can be reduced with minimal effort. PSO algorithms are least expensive, easy, and simpler to apply, and also it has a very good convergence rate. PSO greatly helps in improving the efficiency of the workflow scheduling in cloud environments.

9.3.3 Hybrid Cloud Optimized Cost (HCOC) Schedule Algorithm

HCOC algorithm reduces the makespan to fit the desired execution time or deadline within a reasonable cost. This algorithm will decide which resource has to be leased from the public cloud so that the cost of the allocated tasks is minimized. The hybrid cloud arises when both the public and the private clouds are merged together. In such hybrid clouds, the HCOC algorithm is used in such a way that both the cost and execution time are minimized.

Importance of Scheduling Inside Hybrid Clouds
Scheduling plays an important role in managing the jobs and workflow tasks inside the hybrid clouds. In hybrid clouds, some of the tasks are scheduled over resources in private cloud, whereas the remaining tasks for which suitable resources are not available in private cloud are allocated to the resources in public cloud.

Workflow Scheduling Using HCOC Algorithm
The HCOC algorithm is implemented using the following steps:

```
Initial schedule:
  Schedule the workflow within the private cloud.
    While the makespan is greater than the deadline:
      Tasks are selected for reschedule.
      The resources are selected from the public cloud to compose it to the
        hybrid cloud say H.
    Then reschedule the selected tasks in H.
```

The two main steps in this algorithm are selection of the tasks for rescheduling and selecting the resources from the public cloud to compose it to the hybrid cloud. HCOC is a multi-core-aware algorithm which is used for cost-efficient scheduling of multiple workflows [9]. This algorithm is more efficient when compared to the others because it supports multiple workflows. It mainly follows the QoS parameters for the users which include cost optimization within the user-specified deadline and user-specified budget [9].

9.3.4 Ant Colony Optimization (ACO) Algorithm

ACO is a heuristic algorithm and it is based on the foraging behavior of ant colonies. Initially the ants will be searching their food in random directions. When the ant finds the path to the food source, a chemical substance called pheromone will be left by the ant while returning back to its nest. The density of the pheromone will evaporate if the path to the food source is much longer. If the pheromone density is high, it indicates that many ants have used this path and hence the next ants will be using this particular path. On the other hand, if the pheromone density is small, it means that the path to the food source is much longer and few ants have used this path and this path will be rejected. The path with the highest density pf pheromone will be selected as the optimal solution.

Ant Colony Optimization (ACO) in Workflow Scheduling

In cloud scheduling each path will represent a cloud schedule. One of the main steps to be focused here is the local pheromone update and the global pheromone update.

ACO Algorithm

```
Step 1:   Collect information about the tasks (n) and virtual machines (m)
Step 2:   Initialize expected time to compute (ETC) values
Step 3:   Initialize the following parameters:
          Step 3.1:   Set $\alpha = 1$, $\beta = 1$, $Q = 100$, pheromone evaporation rate ($\rho$) = 0.5, and number of ants = 100.
          Step 3.2:   Set optimal solution = null and epoch = 0
          Step 3.3:   Initialize pheromone trial value $\tau^{ij} = c$.
Step 4:   Repeat until each ant k in the colony finds VM for running all tasks:
          Step 4.1:   Put the starting VM in $tab^{uk}$.
          Step 4.2:   Calculate the probability for selecting VM.
Step 5:   Calculate the makespan for the schedule and select the best schedule based on makespan.
Step 6:   Update pheromone trial value:
          Step 6.1:   Compute the quantity of pheromone deposited.
Step 7:   If the maximum epoch is reached, then the optimal schedule is equal to schedule with optimal makespan.
Step 8:   Go to Step 4.
```

9.3.5 Customer-Facilitated Cost-Based Scheduling in Cloud (CFSC)

The main objective is to minimize the total monetary cost and to balance the load. Both the service providers and the customers will receive the economical benefits only if the resources are properly scheduled. This is mainly done by the cost and the data scheduler in the workflow systems.

The following assumptions are made in the CFSC algorithm:

- A set of heterogeneous virtual machines (VMs) denoted by M are considered for creating cloud environment.
- The communication network is always connected.
- Tasks are executed normally and there are no failures.
- Tasks are non-preemptive.

CFSC Algorithm

Step 1: Compute Rank^u b-level (the length of a longest path from a node to the exit node) of all the nodes.
Step 2: Arrange the nodes in a list by decreasing order of Rank^u values.
Step 3: Arrange the virtual machine list by pricing, in ascending order.
Step 4: Calculate MEFT value for all the nodes.
Step 5: Repeat Steps 6.1, 6.2, and 6.3.
Step 6: Begin:

- **Step 6.1:** The first task vi in the list is removed.
- **Step 6.2:** Find the MEFT value of task vi for all virtual machines.
- **Step 6.3:** Find the vm^j which has minimum MEFT value for task vi, and assign it to vm^j until all the nodes in the list are scheduled.

The CFSC minimizes the cost by using easier cost function algorithm. This CFSC algorithm can be simulated in various cloud tools like Montage, LIGO1, and LIGO 2. Using this kind of meta-heuristic algorithm, the makespan can be minimized.

9.3.6 Task Selection Priority and Deadline (TPD) Algorithm

This algorithm addresses major challenges in cloud computing. Scheduling in cloud can be categorized into the following major types:

- User validation and filtering
- User selection
- Task submission process

User Selection

This is the component selection process based on some behavior or attribute. Here the task of similar type will be selected and scheduled collectively. For cost-based scheduling, the users will have to follow the following steps:

For each user request:

Step 1: Compute start time of each user.
Step 2: The user with minimum cost time will be selected and scheduled.
Step 3: Check the count if the users are having same task time.
Step 4: For the users with the same count, cost-based count will be used.
Step 5: The user with the maximum cost-based time will be selected and scheduled.
Step 6: If the cost-based count is also the same, use the id and update the task.
Step 7: The time limit is not needed since it is a time-based cost.

It is observed that this algorithm is efficient for both the users and the service providers.

Workflow scheduling is one of the major tasks in cloud computing. This section has surveyed the various workflow scheduling algorithms based on cost which are consolidated in Table 9.1. In order to achieve better results, the algorithms should be designed in such a way that it focuses on both the execution time and also the cost. Moving workflows to the cloud computing helps us in using various cloud services to facilitate workflow execution [10].

9.4 Energy-Based Scheduling Algorithms

In the cloud system, these computing resources are provisioned as virtual machines. The VMs are generally provided with various specifications and configuration parameters that include the number of CPU cores, the amount of memory, the disk capacity, etc. The tasks in a workflow are interdependent and must wait for data from their predecessor tasks to continue the execution. Therefore some computing resources are inevitably idle during different time slot, which certainly decreases resource utilization of cloud data centers. This low resource utilization leads to energy waste.

The energy utilization of a cloud platform has focused much in recent years as the scientific workflow executions in cloud platforms acquire huge energy consumption. The energy utilized by the cloud resources assigned for the execution of the workflow (i.e., the virtual machines) can be classified into three components, namely, the processing energy, the storage energy, and the data communication energy. The processing energy is the energy consumed by the virtual machine to execute a specific task. The storage energy represents the energy needed to store data on a permanent storage device (disk memory). The communication energy is

Table 9.1 Cost and deadline-based cloud scheduling survey

S.No	Year/author/journal	Algorithm	Algorithm type	Objective/ metrics	Simulation/ experimental	Experimental scale	Results compared	No. of parameters
1.	2014/Suraj Pandey et al., Linlin Wu, Siddeswara Mayura Guru, Raj Kumar Buyya/Online Technical Publications	Particle Swam Optimization algorithm	Heuristic	Cost and data transmission cost	CloudSim/–	Different scientific workflows	Greedy-S	2
2.	2014/Bittencourt, Luiz Fernando Madeira, Edmundo Roberto Mauro	Hybrid Cost Optimization Cost Scheduling algorithm	Heuristic	Cost, deadline	CloudSim/–	Workflows from different scientific areas: Montage, LIGO, SIPHT, and CyberShake	EHEFT	1
3.	2015/Srinivasan Selvaraj, Jaquline. J/*International Journal of computer Technology and Applications*	Ant Colony Algorithm	Heuristic	Cost, time, deadline	CloudSim	CloudSim	HEFT	3
4.	2015 D.I. George Amalarethinam, T.Lucia Agnes Beena	Customer Facilitated Cost-Based Scheduling in cloud	Meta-heuristic	Cost	CloudSim/–	Montage workflow with 4006 tasks, LIGO 1 workflow/with 2252 tasks, LIGO 2 workflow contains 4000 tasks	HEFT	1
5.	2014/Hong He and Dongbo Liu,/*International Journal of Future Generation Communication and Networking*	An efficient TPD scheduling algorithm for cloud computing	Heuristic	Cost, deadline	CloudSim	20 computing nodes and 7 storage nodes, XCP with version 1.1	HEFT, MMF-DVFS	2

the energy rate needed to transfer data from a virtual machine to another one using a network bandwidth.

Moreover there is a tremendous amount of energy expended in a cloud data center in order to run servers, cool fans of processors, console, monitors, network peripherals, light systems, and cooling system. Mainly the energy consumption of data centers usually causes a large amount of CO_2 emission. In addition to negative environmental implications, increased power consumption may lead to system failures caused by power capacity overload or system overheating. The workflow scheduling algorithm plays a major role for the successful execution of tasks in cloud environments. It is a method which assigns and controls the execution of interdependent tasks on the distributed resources. It maps appropriate resources to workflow tasks in a way to complete their execution to satisfy the objective functions imposed by users. Scheduling is important to make the maximum utilization of resources by appropriate assignment of tasks to the available resources like CPU, memory, and storage. It is highly necessary to devise efficient scheduling algorithms to overcome the energy issues in clouds. This section mainly focuses on the different energy-efficient workflow scheduling algorithms that have been developed to minimize the energy consumption.

Energy management in clouds has been one of the important research interests in recent times. A large number of energy-aware workflow scheduling algorithms are in existence for executing workflow applications in clouds. Generally, these scheduling algorithms can be divided into two categories: meta-heuristic approaches and heuristic approaches.

Xiaolong Xu et al. [11] proposed an energy-aware resource allocation method, named EnReal, to address the energy consumption problem while deploying scientific workflows across multiple cloud computing platforms. This method focuses on the optimal dynamic deployment of virtual machines for executing the scientific workflows in an energy-aware fashion. The energy problems are brought to light as it is estimated that cloud data centers cause about 2% of global CO_2 emission.

The energy consumption in the cloud environment is divided into two major categories, i.e., energy consumption for application execution and energy consumption for dynamic operations. The energy consumption for application execution includes physical machine (PM) baseline energy consumption, the energy consumed by active virtual machines (VMs), idle VMs, and internal communications and external communications between the VMs. The energy consumption for dynamic operations refers to the PM mode switch operations. If all VMs on a single PM are unused, the PM may be put into two modes, i.e., the low-power mode and the sleep mode based on the service time for hosting the allocated applications.

The EnReal method mainly takes into consideration these two types of energy consumptions and aims at reducing both. CloudSim framework is used to evaluate the performance of the energy-aware resource allocation method.

To demonstrate the performance of the proposed EnReal method for scientific workflow executions, it is compared with a baseline method named BFD-M and two existing state-of-the-art methods, i.e., the greedy task scheduling algorithm and the energy-aware greedy algorithm. There are two main focuses in the experimental comparison, i.e., resource utilization and energy consumption. The EnReal method achieves more energy savings by employing both migration-based resource allocation and physical machine mode switch operations.

Zhongjin Li et al. [12] designed a cost- and energy-aware scheduling (CEAS) algorithm with an intent to minimize the energy and cost for deadline constrained, data-intensive, computation-intensive big data applications that require many hours to execute. The energy-related research has gained importance as the cloud providers spent 50% of the management budget for powering and cooling the physical servers and the growing environmental issues like global warming due to CO_2 emissions.

The CEAS algorithm consists of five sub-algorithms. First, the VM selection algorithm is used to map each task to the optimal VM types to reduce the monetary cost. Then, two tasks merging methods such as sequence task merging and parallel task merging methods are employed to reduce execution cost and energy consumption of workflow. Further, the VM reuse policy is used in order reuse the idle time of leased VM to execute the current executable tasks. Finally, the task slacking algorithm is used to reclaim slack time by lowering the voltage and frequency of leased VM instances to reduce energy consumption.

The performance of the CEAS algorithm is evaluated using CloudSim. The four different scientific workflows such as Montage, LIGO, SIPHT, and CyberShake are taken for experimental analysis. Performance comparison of CEAS algorithm is done with three different algorithms. Heterogeneous Earliest Finish Time (HEFT) algorithm, enhanced energy-efficient scheduling (EES) algorithm, and enhancing HEFT (EHEFT) algorithm are used to demonstrate the energy-saving performance of the proposed algorithm.

The comparison results show (refer to Table 9.2) that CEAS algorithm reduces the monetary cost by finding the proper VM using the task merging algorithm. The VM reuse algorithm is used by many tasks to reuse the idle VM instances which help in reducing the energy consumption.

Sonia Yassa et al. [13] suggest a new method to minimize the energy consumption while scheduling workflow applications on heterogeneous computing systems like cloud computing infrastructures. It also concentrates on the two other Quality of Service (QoS) parameters such as deadline and budget. The algorithm devised in this work is a discrete version of the multi-objective particle swarm optimization (MOPSO) combined with Dynamic Voltage and Frequency Scaling (DVFS) technique. The DVFS technique is used to minimize energy consumption by allowing the processors to operate in different voltage supply levels by sacrificing clock frequencies.

The experimental evaluation is done using the pricing models of Amazon EC2 and Amazon CloudFront. The algorithm is compared against HEFT heuristic. The results show that the proposed DVFS-MODPSO is able to produce a set of solutions named Pareto solutions (i.e., nondominated solutions) enabling the user to select the desired trade-off.

Khadija Bouselmi et al. [14] proposed an approach that focuses on the minimization of network energy consumption, as the network devices consume up to one-third

Table 9.2 CEAS algorithm summary [12]

Objective	VM selection	Sequence task merging	Parallel task merging	VM reuse	Task slacking
Reduce cost	–	Yes	Yes	Yes	–
Reduce energy	–	Yes	Yes	Yes	Yes
Time complexity	O(Kn^2(n+e))	O(Kn)	O(n^{2}logn)	O(nL)	O(n^2)

of the total energy consumption of cloud data centers. There are two major steps involved in this process.

In the first step, a Workflow Partitioning for Energy Minimization (WPEM) algorithm is utilized for reducing the network energy consumption of the workflow and the total amount of data communication while achieving a high degree of parallelism. In the second step, the heuristic of Cat Swarm Optimization is used to schedule the generated partitions in order to minimize the workflow's overall energy consumption and execution time.

The simulation environment used for evaluation is the CloudSim toolkit. Two other workflow partitioning algorithms are implemented to evaluate and compare the test results of WPEM approach. The first algorithm denoted as Heuristic 1 is the multi-constraint graph partitioning algorithm. To limit the maximum number of partitions generated by Heuristic 1, another version of it, denoted as Heuristic 2, which sets the maximum number of partitions, is implemented.

The performance evaluation results suggest that the proposed WPEM algorithm allows reducing remarkably the total energy consumption and the additional energy consumption incurred from the workflow partitioning, by reducing the data communication amount between the partitions and reducing the number of used VMs.

The authors, Peng Xiao et al. [15], present a novel heuristic called Minimized Energy Consumption in Deployment and Scheduling (MECDS) for scheduling data-intensive workflows in virtualized cloud platforms. This algorithm aims at reducing the energy consumption of intensive data accessing. The data-intensive workflow will generate a large volume of intermediate data, which requires being stored in independent storage nodes. When running a data-intensive workflow, input data of an activity node is transferred from an independent storage node to the execution node, and output data is transferred back to the original storage node or others. Such an interweaving makes it more complex for scheduling data-intensive workflows.

The algorithm consists of two phases such as VM deployment and DAG scheduling. In the phase of VM deployment, physical resources are mapped into a number of VM instances. So, the original power model of a physical machine should be translated into the VM-oriented power model. In the phase of DAG scheduling, activities in the workflow are assigned onto a set of VM instances; therefore, the total energy consumption is dependent on the VM power models and the execution time of each VM.

The experimental results in comparison with the algorithms such as MMF-DVFS, HEFT, ECS+idle, and EADAGS show that the proposed algorithm is more robust than other algorithms, when the system is in the presence of intensive data-accessing requests.

Huangke Chen et al. [16] propose an energy-efficient online scheduling algorithm (EONS) for real-time workflows. This algorithm is used to improve the energy efficiency and provide high resource utilization. The major contribution of this work is to schedule the workflows to VMs while improving VMs' resource utilization and ensuring the timeliness of workflows. It also emphasizes on scaling up/down the computing resources dynamically with the variation of system workload to enhance the energy efficiency for cloud data centers.

CloudSim toolkit is used for the experimentation. The performance of this algorithm is compared with three other algorithms such as EASA, HEFT, and ESFS. The experimental results show that EONS achieves a better performance in terms of energy saving and resource utilization while guaranteeing the timing requirements of workflows.

Guangyu Du et al. [17] aim at implementing an energy-efficient task scheduling algorithm, as scheduling plays a very important role in successful task execution and energy consumption in virtualized environments. The energy issues are taken into consideration in this work as large numbers of computing servers containing virtual machines of data centers consume a tremendous amount of energy.

The proposed algorithm is comprised of two main objectives. The first objective is to assign as many tasks as possible to virtual machines with lower energy consumption, and the second objective is to keep the makespan of each virtual machine within a deadline. The experimental evaluation is done using CloudSim toolkit, and the results compared against HEFT, GA, and HPSO show that there is effective reduction in energy consumption and the tasks are completed within the deadline.

Zhuo Tang et al. [18] formulated a DVFS-enabled energy-efficient workflow task scheduling (DEWTS) algorithm. This algorithm is mainly used to achieve energy reduction and maintain the quality of service by meeting the deadlines. This algorithm initially calculates the scheduling order of all tasks and obtains the whole makespan and deadline based on Heterogeneous Earliest Finish Time (HEFT) algorithm. Then by resorting the processors with their running task number and energy utilization, the underutilized processors can be merged by closing the last node and redistributing the assigned tasks on it.

Later, in the task slacking phase, the tasks can be distributed in the idle slots by leveraging DVFS technique.

The experimental evaluation is done using CloudSim framework, and the proposed algorithm is compared against EES and HEFT algorithm. Compared to the other two algorithms, DEWTS provides better results as the tasks can be distributed in the idle slots under a lower voltage and frequency, without violating the dependency constraints and increasing the slacked makespan.

Yonghong Luo and Shuren Zhou [19] describe a power consumption optimization algorithm for cloud workflow scheduling based on service-level agreement (SLA). The main objective of this work is to reduce power consumption while meeting the performance-based constraints of time and cost. This algorithm works by searching for all feasible scheduling solutions of cloud workflow application with critical path, and then the optimal scheduling solution can be found out by calculating total power consumption for each feasible scheduling solution.

The authors, Hong He and Dongbo Liu [20], presented a new approach to deal with the data-accessing related energy for the data-intensive workflow applications that are executed in cloud environment, as several studies focus on CPU-related energy consumption. A novel heuristic named Minimal Data-Accessing Energy Path (MDEP) is designed for scheduling data-intensive workflows in virtualized cloud platforms. The proposed scheduling algorithm consists of two distinguished phases: firstly, it uses MDEP heuristic for deploying and configuring VM instances with the aim to reduce the energy consumption spent on intermediate data accessing; secondly, it schedules workflow activities to the VM instances according to VM power model. This method uses a conception called Minimized Energy Consumption in Deployment and Scheduling (MECDS). This is used to select a storage node aiming to obtain minimal data-accessing energy consumption for the current activities.

The experiments performed mainly focus on the characteristic of workflow energy consumption and execution performance. The experimentation is done in the CloudSim environment. The algorithm is compared against HEFT, MMF-DVFS, ECS+idle, and EADAGS. To further investigate the energy-efficiency, three measurements are introduced: Effective Computing Energy Consumption (ECEC), Effective Data Accessing Energy Consumption (EDAEC), and Ineffective Energy Consumption (IEEC).

The results show that the proposed work can significantly reduce the energy consumption of storing/retrieving intermediate data generated during the execution of data-intensive workflow and offer better robustness.

Table 9.3 provides an insight on the various energy-efficient algorithms discussed above.

9.5 Automated Workload Consolidation

Distributed business workloads/applications need to be consolidated in one place to achieve better performance. For example, multi-tiered applications give higher throughput if their contributing components stick together. Similarly there are several workloads and a myriad of parameters to decide whether they have to be together to provide higher productivity. At the infrastructure level, there are automated mechanisms for virtual machine (VM) placement, live-in migration across data centers, etc. in order to suitably place workload-hosted VMs to guarantee the SLA agreed between providers and consumers. There are techniques and tools emerging and evolving fast in order to ensure automated workload consolidation and optimization through appropriate VM placement at runtime. Apart from computational efficiency, energy efficiency is emerging as the viable factor for providing automated workload consolidation. There are well-intended approaches and algorithms for speeding up the process of workload optimization without any risks.

Table 9.3 Energy-aware cloud scheduling survey

S.No	Year/author/journal	Algorithm	Algorithm type	Objective/metrics	Simulation/ experimental	Experimental scale	Results compared	No. of Parameters
1.	Xiaolong Xu, Wanchun Dou, Xuyun Zhang, and Jinjun Chen, Senior Member, IEEE Transactions on Cloud Computing	EnReal – Energy-aware Resource Allocation method	Heuristic	Energy	CloudSim	Number of scientific workflows {50,100,150,200,250,300}/ Number of tasks in each workflow [4, 25]/Numbers of required VMs for each task [1, 14]	Greedy-S, Greedy-D, BFD-M	1
2.	Zhongjin Li, Jidong Ge, Haiyang Hu, Wei Song, Hao Hu, and Bin Luo/IEEE Transactions on Services Computing	Cost and Energy Aware Scheduling (CEAS) Algorithm	Heuristic	Energy, cost, deadline	CloudSim	Workflows from different scientific areas: Montage, LIGO, SIPHT, and CyberShake./VM types – 10, bandwidth between VM instances is a constant 1GB	EES, EHEFT	3
3.	Sonia Yassa, Rachid Chelouah, Hubert Kadima, and Bertrand Granado/Hindawi Publishing Corporation	Dynamic Voltage and Frequency Scaling, Multi-Objective Discrete Particle Swarm Optimization	Heuristic	Energy, makespan, cost	Amazon EC2, Amazon CloudFront	Swarm size = 50 particles and the maximum number of iterations = 100	HEFT	3

4.	2016/Khadija Bouselmi, Zaki Brahmi, Mohamed Mohsen Gammoudi/2016 IEEE International Conference on Services Computing	Workflow Partitioning for Energy Minimization (WPEM) algorithm		Energy,/execution time	CloudSim	Montage workflow with 4006 tasks, CyberShake workflow/with 2252 tasks, Epigenomics workflow contains 4000 tasks	NA	2
5.	2013/Peng Xiao, Zhi-Gang Hu, Yan-Ping Zhang/*Journal of Computer Science And Technology*	Minimized Energy Consumption in Deployment and Scheduling (MECDS)	Heuristic	Energy	Xen Cloud Platform	20 computing nodes and 7 storage nodes, XCP with version 1.1	HEFT, MMF-DVFS, ECS+idle, EADAGS	1
6.	2016/Huangke Chen, Xiaomin Zhu, Dishan Qiu, Hui Guo, Laurence T. Yang, Peizhong Lu/2016 45th International Conference on Parallel Processing Workshops	EONS algorithm	Heuristic	Energy and resource utilization	CloudSim	CPU resource requirements/(in MHz) range from 200 to 2000 with an increment of 200./The inter-VM bandwidth is assumed to be 1 Gbps. The start-up time of a host is 60s and the creation time of a VM is 30s	EASA, HEFT, ESFS	2

(continued)

Table 9.3 (continued)

S.No	Year/author/journal	Algorithm	Algorithm type	Objective/metrics	Simulation/ experimental	Experimental scale	Results compared	No. of Parameters
7.	2014/Guangyu Du, Hong He, and Qinggang Meng/ Hindawi Publishing Corporation			Energy, deadline	CloudSim	RAM size for all the VMs is set to 512MB. The number of the/VM set is fixed at 12. The speed of each VM (MIPS) is chosen/uniformly in [102, 103]	HEFT, GA, HPSO	2
8.	2015/Zhuo Tang, Ling Qi, Zhenzhen Cheng, Kenli Li, Samee U. Khan, Keqin Li/Springer	DVFS-enabled energy-efficient workflow task/ scheduling algorithm	Heuristic	Energy, deadline	CloudSim	The number of random DAG tasks: {20, 40, 80,/160, 320, 400}. The communication to computation ratio (CCR) set: {0.1, 0.5, 1.0, 2.0, 5.0}. The set of processors available to use is from 2 to 32	HEFT, EES	2
9.	2013/Yonghong Luo, Shuren Zhou/ WSEAS Transactions on Systems	Power Consumption Optimization Strategy of Cloud Workflow Scheduling Based on SLA		Energy, time, cost	CloudSim	Time constraint (Montage: 400s, Epigenomics: 400s,/ MRI: 350s, e-protein: 350s) and a cost constraint/ (Montage: 35$, Epigenomics: 35$, MRI: 30$,/e-protein: 30$) for each cloud workflow application	Deadline-MDP, Loss and Gain, PCOA	3
10.	2014/Hong He and Dongbo Liu,/*International Journal of Future Generation Communication and Networking*	Minimal Data-Accessing Energy Path/ for scheduling data-intensive workflow	Heuristic	Energy	CloudSim	Bandwidth – 10MB to 1 GB, CCR – 0.1 to 10.0	HEFT, MMF-DVFS, ECS+idle, EADAGS	1

9.6 Conclusion

In this chapter, a comprehensive survey of various existing workflow scheduling algorithms that focuses on minimization of energy consumption is presented. The issues and challenges of scheduling in the cloud computing environment are also discussed extensively. Several algorithms are analyzed to determine the energy efficiency achieved through the proposed methods. The comparison of each method with other existing algorithms is also highlighted. Reduction of energy consumption is done mainly by improving the resource utilization. Dynamic Voltage Scaling (DVS) has been also proven to be an effective technique for energy savings. More efficient algorithms can be developed to improve the energy efficiency for executing the workflows in cloud environment. Several workflow optimization approaches are explained in Chap. 10.

References

1. Kataria D, Kumar S (2015) A study on workflow scheduling algorithms in cloud. Int J Res Appl Sci Technol 3(8):268–273
2. Tao J, Kunze M, Rattu D, Castellanos AC (2008) The Cumulus project: build a scientific cloud for a data center. In Cloud Computing and its Applications, Chicago
3. Chopra N, Singh S (2014) Survey on scheduling in Hybrid Clouds. In: 5th ICCCNT–2014 July 1113, Hefei, China
4. Abawajy JH (2004) Fault-tolerant scheduling policy for grid computing systems. In: Proceedings of parallel and distributed processing symposium, 2004, 18th international, IEEE, p 238
5. Selvarani S, Sudha Sadhasivam G (2010) Improved cost based algorithm for task scheduling in cloud computing. In: IEEE
6. Abrishami S, Naghibzadeh M, Epema DH (2012) Cost-driven scheduling of grid workflows using partial critical paths. IEEE Trans Parallel Distrib Syst 23(8):1400–1414
7. Choudhary M, Sateesh Kumar P (2012) A dynamic optimization algorithm for task scheduling in cloud environment. 2, 3, pp.2564-2568
8. Calheiros RN Ranjan R, De Rose CAF, Buyya R (2009) Cloudsim: a novel framework for modeling and simulation of cloud computing infrastructures and services, Arxiv preprint arXiv:0903.2525
9. George Amalarethinam DI, Joyce Mary GJ (2011) DAGEN–A tool to generate arbitrary directed acyclic graphs used for multiprocessor scheduling. Int J Res Rev Comput Sci (IJRRCS) 2(3):782
10. Jangra A, Saini T (2013) Scheduling optimization in cloud computing. Int J Adv Res Comput Sci Softw Eng 3(4)
11. Xiaolong Xu, Wanchun Dou, Xuyun Zhang, Jinjun Chen (2016) EnReal: an energy-aware resource allocation method for scientific workflow executions in cloud environment. IEEE Trans Cloud Comput 4(2):166–179
12. Zhongjin Li, Jidong Ge, Haiyang Hu, Wei Song, Hao Hu, Bin Luo Cost and energy aware Scheduling algorithm for scientific workflows with deadline constraintin clouds. In: IEEE Transactions on Services Computing. doi:10.1109/TSC.2015.2466545
13. Yassa S, Chelouah R, Kadima H, Granado B (2013) Multi-objective approach for energy-aware workflow scheduling in cloud computing environments. Hindawi Publishing Corporation. ScientificWorld J 2013, Article ID 350934:13. http://dx.doi.org/10.1155/2013/350934

14. Bouselmi K, Brahmi Z, Gammoudi MM (2016) Energy efficient partitioning and scheduling approach for scientific workflows in the cloud. In: IEEE international conference on services computing, doi:10.1109/SCC.2016.26
15. Peng Xiao, Zhi-Gang Hu, Yan-Ping Zhang (2013) An energy-aware heuristic scheduling for data-intensive workfows in virtualized datacenters. J Comput Sci Technol 28(6):948–961. doi:10.1007/s11390-013-1390-9
16. Huangke Chen, Xiaomin Zhu, Dishan Qiu, Hui Guo, Laurence T. Yang, Peizhong Lu (2016) EONS: minimizing energy consumption for executing real-time workflows in virtualized cloud data centers, 2332-5690/16 $31.00 © 2016 IEEE DOI 10.1109/ICPPW.2016.60
17. Guangyu Du, Hong He, Qinggang Meng (2014) Energy-efficient scheduling for tasks with Deadline in virtualized environments. Math Probl Eng 2014, Article ID 496843: 7. http://dx.doi.org/10.1155/2014/496843
18. Zhuo Tang, Ling Qi, Zhenzhen Cheng, Kenli Li, Samee U. Khan, Keqin Li (2015) An energy-efficient task scheduling algorithm in DVFS-enabled cloud environment, doi 10.1007/s10723-015-9334-y, © Springer Science+Business Media Dordrecht.
19. Yonghong Luo, Shuren Zhou (2014) Power consumption optimization strategy of cloud workflow scheduling based on SLA. WSEAS Trans Syst 13: 368–377, E-ISSN: 2224-2678
20. He H, Liu D (2014) Optimizing data-accessing energy consumption for workflow applications in clouds. Int J Future Gener Commun Netw 7(3):37–48

Chapter 10
Automated Optimization Methods for Workflow Execution

Abstract Workflow optimization is an approach to enhance the speed, robustness, and compactness of workflows by exploiting their structure, runtime, and output. This chapter initially highlights the significance of workflow optimization along with different possible levels of optimization. Further, it outlines the Taverna optimization framework over single and distributed infrastructure together with the optimization plug-ins that are validated using two scientific workflow executions.

10.1 Introduction

Scientific workflows are emerged as a key technology that assists the scientists in performing their experiments due to the factors that it offers, like higher level of abstraction, automation, comprehensive, and a graphical alternative for the existing script-based programming. Additionally, it can also be used for recreating the entire work without the need for recreating the experiment. However, workflows rely on several technical choices that lay out the foundation to optimize toward speedup, robustness, and compact to the scientists. But the objective of optimization has different focuses, such as topology, runtime, and output optimization. Engineering frameworks offer large variety of optimization algorithm, and hence user must test different search methods and apply the improved and novel techniques. Hence developers are in need for a platform that offers abstraction and general mechanisms to provide workflow optimization for different algorithms and levels.

Section 10.2 illustrates the significances of workflow optimization during workflow execution. Additionally, it also highlights some of state-of-the-art optimization approaches that are practiced in the literature.

Section 10.3 portrays the inclusion of an optimization phase in the traditional workflow life cycle. Besides, it outlines the several possible optimization levels such as parameter, component, topology, runtime, and output optimizations.

Sections 10.4 and 10.5 explain the execution of Taverna optimization framework over single and distributed infrastructure. The workflow parameter optimization techniques such as genetic algorithm, ant colony optimization, and Particle Swarm Optimization are explained in Sect. 10.6. The detailed control flow of optimization plug-in is described in Sect. 10.7. Following that, Sect. 10.8 validates the

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_10

optimization plug-in using two scientific workflows: proteomics and biomarker. Finally, Sect. 10.9 consolidates the concluding remarks for the chapter.

10.2 Workflow Optimization: Current State of the Art

Scientific workflows are complex and computationally intensive in nature. Their execution time might range from few minutes to several hours. For given problem, many different workflow instances must be executed, which in turn reduces the execution time for optimization. Further, the acceleration strategies should also be employed like parallel processing and moving the time-consuming and parallel task to improve the performance of the computing resources. Concept of workflow also depends on technical preferences and established the aspects that make optimization, in terms of fastness and robustness, handy to the scientists. Domain scientists, who don't have experience on computer usage, stick with their custom technologies which is a good reason that the scientists are not applying existing optimization workflows. Many designed frameworks that use programming interfaces or configuration files, for setting up the optimization process. Also their complexity extends as it undergoes optimization process, which is in contrast to the requirement that the complexity must be hidden from the user and the main function is optimizing and not setting up configurations or design another workflow that performs the optimization. Hence to deal with these issues, robust and versatile optimization techniques, such as *heuristic algorithms*, are required. Compared with the parameter sweep, meta-heuristic algorithms explore a smaller part of the parameter search space. Additionally, it also supports parallel mechanism. *Hence workflows must be handled by the heuristic search methods*.

Generally, the optimization can be explained as systematic comparison of all the possible parameter sets and thus finding the unknown parameter set, which is the best set among all those ones. The optimization should focus on either maximizing or minimizing their objective functions or fitness functions. The applications can be in a wide range of fields, for example, right from modeling the aircraft wings [1] to drug discovery applications [2]. And for this wide range of applications, their objective/fitness function also varies in corresponding to their needs. Optimization problems can be single- or multi-objective optimization problems [3]. Solution for the problem can be local or global optimum. By local, we mean that the optimum valve which is needed is available, and, by global, there is no optimum value for that function itself.

In addition to these, the optimization problems can be either linear or nonlinear. The scientific workflow problems can be available as both linear and nonlinear. Also most of the life science algorithms are nonlinear problems, since their objective functions are not continuous and differential with respect to decision variables. These nonlinear optimization problems can be solved by various optimization algorithms, which sample the search space to find the good or optimal solution that too at reasonable computation time.

For single-objective optimization, several methods like deterministic, stochastic, and meta-heuristic method can be used [4]. Optimizing the single-objective problems results in one-dimensional vector of values called as *fitness values*. The deterministic algorithms are iterative methods that move toward the optimal value in each iteration or step. Examples for these deterministic algorithms are gradient descent [5, 6], branch and bound [7], etc. Simulated annealing (SA) [8] is a stochastic process, but, sometimes, it is mentioned as heuristic also. SA aims at finding the optimum valve via simulating the cooling process of materials.

Heuristic search methods [9–11] are widely used in bioinformatics [12] since it aims at seeking the optimal solution at the reasonable time. The examples for heuristic algorithms are evolutionary algorithm (EA), that too in particular genetic algorithm (GA) [13, 14], Particle Swarm Optimization (PSO) [15, 16], and ant colony algorithm (ACO). EA simulates the process of biological evolution by selection, crossover, and mutation to reproduce the individuals and select the best among them. PSO moves around the candidate solution, which is known as particle, in an n-dimensional search space. ACO was originally designed for discrete optimization problems, but they have adapted to the continuous problems also. They produce the populations of ants that represent the several possible solutions, which can be changed according to pheromone density. It is adopted by each ant at every step for better efficiency.

Often the multiple objects will be considered in optimization process. For example, the trade-off between sensitivity-specificity and result quality-runtime is related, and hence one affects the other. For these factors, the multi-objective optimization is needed. For the multi-objective optimization, the minimization or maximization of the optimal result is represented by the multidimensional vector of values, which is known as Pareto optimal set (PS). The image of PS at the objective function space is called as Pareto front (PF). PS has number of solutions, in which there will be compromise between the different objective functions, and hence there is no best solution for the given condition. Decision-maker can be used either at optimization or after the optimization. It selects a set of optimal solutions, since no single solution is optimal.

Multi-objective optimization (MO) problems can be commonly solved via multi-objective evolutionary algorithm (MOEA) [17–19]. A challenge for MO algorithm is rating of the different solutions. In MO algorithm, each solution must be rated, whose rating is regarding non-dominance and closeness to PF. The approach for PSO, SA, and other meta-heuristic multi-objective algorithms has been developed. But the fact is they cause several more challenges than MOEA. In addition to this, the identification of global and local best particles is difficult. Solving real-time problems, like molecular docking [20], structure activity relationship [21], phylogenetic tree interferences [22], drug discovery [2], etc., is still crucial and challenging, due to their complex adaptation to the appropriate problem model. *Hence multi-objective optimizations are not generally used in life sciences*. Similarly, MO optimization is not widely used for parameter optimization.

Within single- or multi-objective optimization, the hybrid approaches [23–25] are popular as it utilizes characteristics of both their advantages. Primarily, the deterministic and stochastic algorithms are applied to single-objective optimization. One such approach is called *integrative method* in which one master optimization algorithm uses the other optimization algorithms as a substitute for a specific function alone. Another one is *collaborative method*, in which two or more algorithms run in parallel and do exchange their information.

10.3 Modified Workflow Life Cycle and Optimization Levels

Entire process of experimental investigation is described as scientific workflow life cycle, which tells each stage of evolution right from initial to final. Common scientific workflow life cycle foresees analysis and learning phase, after the execution of the workflow, in order to define the initial design and head toward final result. For smaller workflows, the trial and error method is feasible, while for larger workflows, it is impractical and time-consuming. Re-execution of entire workflow while only parts of workflow need it is resource efficiency affecting factor. To tackle this, the new phase called optimization phase is concatenated with the common life cycle as shown in Fig. 10.1.

Design and refinement phase: Starts from the repository. The components are selected representing the individual steps of experiments. Composition of the components is established.

Planning and sharing phase: In this phase, turning the abstract workflow into concrete executable workflow is done, which is achieved by mapping abstract parts to the concrete applications/algorithms. Parameters and data sources are defined and the execution resources are selected. Thorough planning is needed for large-scale and compute-intensive workflows, since the high-performance computers/

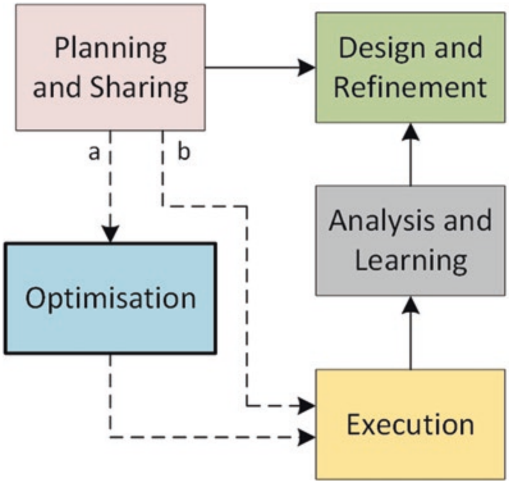


Fig. 10.1 Modified workflow life cycle

grids/cloud resources need precisely defined informations. At last, this is also meant for sharing the design with the community.

Execution phase: Execution is managed by the workflow engine, which maps the executable workflow into appropriate execution environment. This mapping is about retrieving the information about available software, computing resources, data resources, etc. Workflow components are then executed in predefined order, consuming defined datasets with monitoring capability by the engine. The result is fed to engine, which is delivered to the user.

Analysis and Learning Phase: In order to successfully evolve the experiments, this phase is used. It is basically comparing the obtained results with the other silico or in vitro experiments. If the result doesn't match the expected one, then the scientists restart the workflow life cycle.

New optimization phase: This phase is for making it as automated and generic one. In common workflow life cycle, optimization is manually performed within the analysis and learning phase, and hence the entire process has to be re-executed to obtain the refined results. Hence the rearrangement of the phases is required so that it would give acceptable result.

10.3.1 Optimization Levels

The newly introduced optimization phase is not limited to a particular type of optimization. Optimizations can target different levels, namely, parameter optimization, component optimization, and topological optimization. Each optimization level can be achieved by different optimization algorithms.

10.3.1.1 Parameter Optimization

The most common task is improvement in parameters in optimization. The workflows usually combine different data input and several different components, within its parameter, and hence there will be a number of possible parameter optimizations for any workflow. This makes the parameter optimization a difficult task. Optimal parameter combination of a workflow depends on the input data, while some parameters will also have influence on final results. Since parameter sweeps fail when the number of parameters rises, the method that can handle the nonlinear workflow optimization that too at a reasonable time, which can be achieved by exploring smaller part of parameter search space, is needed. Hence the proposal used heuristic optimization algorithm. In order to reduce the search space further, the user's assistance via constraints and dependencies between the parameters are needed. Dependencies could be described as mathematical or logical functions and as a fixed set of allowed values.

10.3.1.2 Component Optimization

Sometimes two or more algorithms implement equivalent function but with different characteristics in such a manner that each one is optimal for each data type of input. However, the component optimization attempts to find the optimal algorithm to fulfill the requirements. For example, the choice of local sequence alignment method can be implemented either by BLAST [26] or by profile alignment methods PSI-BLAST [27]. The user has to select which is best. It also supports the null component. The components with same input and output can be switched on and off, so that their testing efficiency and utility can be achieved. The parameter optimization should be applied in combination with the component optimization by which we say that they can be jointly encoded. Apart from the information obtained from the users, like constraints and applied methods, any other information that is obtained from optimization of similar workflows also helps.

10.3.1.3 Topology Optimization

At some cases, the reorganization of the data and the enactment flow can result in benefits, in order to perform that their output formats must be identical in nature. If the data formats are not compatible with each other, then a new individual component is introduced that transforms the data into appropriate formats. They are known as *shims.*

10.3.1.4 Runtime Performance Optimization

One common target in optimization is to improve the workflow's runtime performance [28]. Typical scientific workflows are executed in e-Science infrastructure; hence several approaches exist to intelligently schedule the workflows or tasks that are on the grid and cloud [29]. Scientific workflow runtime optimization is very heterogeneous since they take different criteria into account. The criteria can be given as follows: workflow structure that aims at clustering the tasks; data processing which optimizes the execution regarding their data usage; and component/task model that schedules the job regarding their task type. In optimization, the other factors like cost, time, and QoS parameters must also be taken into account. Optimization extension for WINGS/Pegasus SNMS, developed by Kumar et al. [30], focuses on optimization of runtime performance via modifying the application parameters. Integrated framework takes QoS requirements into account to adjust data-dependent parameters, so that the distributed data processing of the application can be implemented. Other proposed mechanisms apply heuristic-based optimization algorithm for managing the scheduling. The available solutions target on fixed bi-criteria optimization and incorporate ACO [31], MOEA [32], and GA [33] for scheduling and recursive loop handling. Active harmony provides the performance tuning test that can be used or modified without in-depth knowledge of the domain. Also, this allows users to change the parameters at runtime.

10.3.1.5 Output Performance Optimization

During writing, no comprehensive approach for general workflow optimization was available, except Nimrod/OK [34] which is a parameter optimization tool form engineering domain. Nimrod/OK extends SWMS Kepler [35] with the so-called actors that apply optimization. The optimization process is itself seen as a workflow with number of steps in it. In Nimrod/OK, the user has to define a workflow with a minimum of four tasks, namely, setup of the search space diagram, point generator, optimization process, and optimization reports. The computational model that represents the objective function is defined as plain function expression or MATLAB Nimrod actor execution. If user wants to optimize a scientific workflow, it requires enveloping the workflow (that has to be optimized) into a second specific optimization workflow. By this, user can combine various optimization methods into a single workflow. Another method is developed by Crick [36] that used Taverna management system [37] to design a parameter optimization workflow.

Apart from discussed optimization levels, other levels can also be defined in the future, which sits well for optimization algorithm and is in continuous development process. Users can demand an algorithm where the efforts required by them are kept as small as possible. In addition to this, the developers must also feel free to adapt and reuse the framework, and, for doing so, the convenient frame including the basic stock of functionalities must also be provided. Back ends of optimization have same requirements, namely, several sub-workflows have to be involved, enacted, and monitored, and results must be extracted. These mechanisms rely on Taverna-specific functions.

In addition to these, the users avoid using script languages and demand for user-friendly graphical interfaces, which tell that the interface should use common interfacing elements, be less complex, and have similar look and feel for different optimization algorithms. To avoid creation of GUI each time, it should be decoupled from the framework. Workflow within a workflow must never be used since it is either uncomfortable or user-friendly.

10.4 Taverna Optimization Framework

The optimization framework is implemented on top of any existing workflow management system. Here, the optimization framework is integrated in Taverna. The requirements of the framework are as follows: hide the complexity from user, provide an extensible as well as user-friendly GUI, and implement a mechanism that provides the required functionalities. The architecture of Taverna with the proposed APIs and possible optimization plug-ins is given in Fig. 10.2.

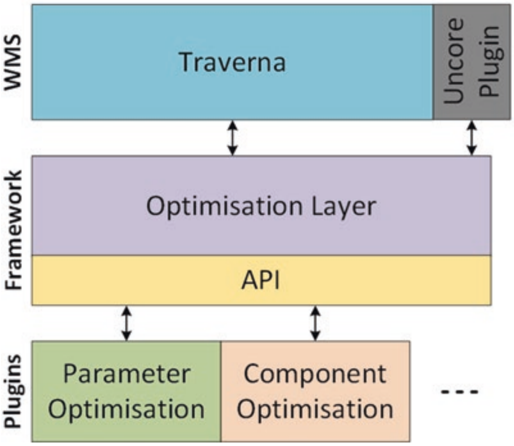


Fig. 10.2 Taverna framework with optimization plug-ins

The framework must offer an API to access the specific functions and GUI to offer user-friendly nature. Taverna is basically not developed for supporting the scientific workflow optimization methods, and hence it has to be developed for that adaptation.

To make the Taverna support workflow optimization, the major requirement is accessibility of all input and output parameters and sub-workflows. By accessibility, we mean the rights to modify the workflow parameter, execute the modified workflow, and evaluate the result. In addition to these, the mechanisms for interrupting the execution of the entire workflow at a specific point execute the sub-workflow several times. To implement these requirements in Taverna, the first approach aims at extending and reusing *service provider interfaces* (SPI) that are provided by Taverna.

One such SPI provides usage of *processor dispatch stack* of Taverna. The processor dispatch stack is a mechanism to manage the execution of a specific activity. Before the activity is invoked, the predefined stack of layers is called from up to bottom, and after that, the execution is from bottom to top. In order to integrate the optimized execution in Taverna, the dispatch stack was extended with a new layer, named as *optimize layer*, on top of the stack. This layer interrupts the process of workflow execution before a specific component is executed.

Additionally, the advantage was taken from the sub-workflow concept, which is provided by Taverna. The sub-workflow concept allows definition and execution of the sub-workflow. This implies that one can manually select a sub-workflow and process it by the dispatch stack. Only the lowest level of the workflow is decomposed into individual activities, which gains traverse the stack itself. Simply the processor dispatch stack provides access to sub-workflows, which in turn provides access to all parameters, data structure, dataflows, etc., during the invocation of optimize layer.

Optimization framework also utilizes a GUI-SPI, and it implements a new uniform perspective into Taverna Workbench. This perspective implements a general

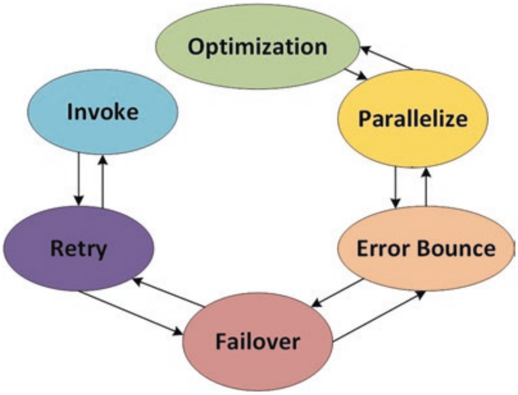


Fig. 10.3 Processor dispatch stack of Taverna

setup of an optimization run. The common workflow diagram and selection pane were arranged within these, by which the users can define sub-workflow, which shall be the subject to the optimization. After the selection, the components are not subjected to optimization will appear as gray in workflow diagram as shown below. Lower left pane shows the specific interface that is implemented by the respective optimization plug-in. To provide the access and usage of the execution model and user interface mechanisms, an API is developed for framework. This API allows the users to extend the optimization method of Taverna's specific execution and GUI functionalities that too without reinventing and re-implementing those.

APIs provide methods for requesting GUI pane, to provide the text fields, for optimization; starting options for specific plug-ins; receiving the modified parameter or sub-workflow to start the execution; receiving the fitness values from the workflow to forward it to the plug-ins; requesting the termination criteria and deciding when optimization is finished; and requesting the best result to return to the users.

In the new perspective optimization, users can select the sub-workflow and optimization-specific informations. After that the newly integrated run button activates the fully automated optimization process. Workflow is executed until the respective sub-workflow is reached, and then the newly embed optimize layer extracts the information and forwards it to the particular optimization plug-in for modifications. Then the plug-in does the proper optimization methods and the sub-workflow refinement, while the framework does nothing during that.

The plug-in returns the set of new sub-workflow entities with set of different parameters. The optimization framework then executes the workflow by utilizing the topmost Taverna layer, *parallelize* (refer to Fig. 10.3). Taverna provides a queue, which is filled with the sub-workflows for parallelize layer and pushes each sub-workflows down the stack in a separate thread. After execution of all sub-workflows, the optimize layer receives the set of results from parallelize layer, which represents the fitness value, and these will be passed to a specific optimization plug-in for analysis and evaluation.

10.5 Optimization Using Distributed Computing Infrastructure

Proposed one schedules several independent executions of the sub-workflows. Since the workflow is complex [38], they should run in parallel to obtain the optimization in a reasonable amount of time. But running several parallel executions may cause bottleneck in workflow engine [39] and not recommended on Taverna client in many cases. Hence the execution is moved into distributed computing resources since it lowers the workload on client.

10.5.1 Three-Tier Execution Architecture

The uppermost tier of Taverna Workbench (refer to Fig 10.4) is established on the user, and in this, the users can set up the workflow, configure the inputs and outputs, and also describe the optimization composition at each generation of optimization process. In order to allow workflow execution in a distributed fashion, UNICORE Taverna plug-in was adapted in framework. Since all the workflow executions are independent, the execution can be done in a parallel manner. Hence, each individual workflow instance is submitted to UNICORE Service Orchestrator (SO).

UNICORE's SO manages distribution of the workflow on computing infrastructure and monitors the status of workflow execution. For distributing, various brokering strategies are used to find the best suited set of resources and also to enable the load balancing. By this, the workflow is submitted in distributed way, and one instance of Taverna Server is utilized.

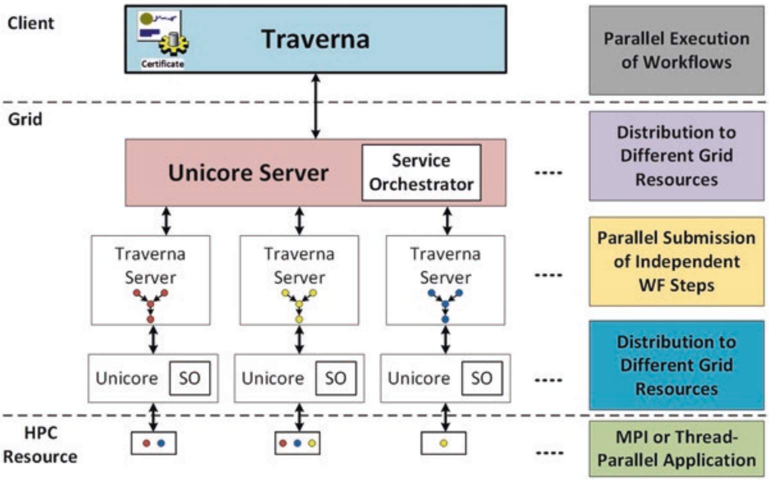

Client
Certificate
Traverna
Parallel Execution of Workflows
Grid
Unicore Server
Service Orchestrator
Distribution to Different Grid Resources
Traverna Server
Traverna Server
Traverna Server
Parallel Submission of Independent WF Steps
Unicore
SO
Unicore
SO
Unicore
SO
Distribution to Different Grid Resources
HPC Resource
MPI or Thread-Parallel Application

Fig. 10.4 Three-tier execution architecture

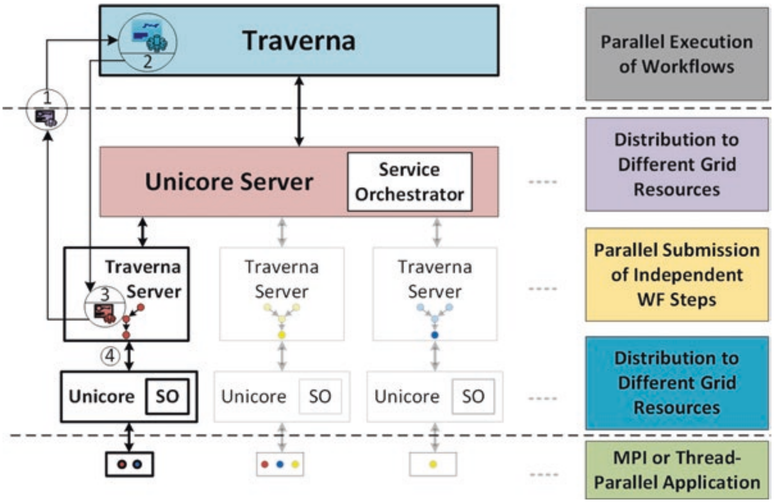

Traverna
1
2
Unicore Server
Service Orchestrator
Traverna Server
3
Traverna Server
Traverna Server
4
Unicore
SO
Unicore
SO
Unicore
SO
Parallel Execution of Workflows
Distribution to Different Grid Resources
Parallel Submission of Independent WF Steps
Distribution to Different Grid Resources
MPI or Thread-Parallel Application

Fig. 10.5 Implementation of trust delegation

In the second tier, the workflow has several independent steps that can be represented as individual UNICORE jobs. These independent steps are submitted for parallel execution, due to the parallel execution model of Taverna and job distributing nature of SO, on the high-performance computing (HPC) resources. In the third tier, the workflow steps are implemented as MPI/thread parallel applications [40].

10.5.2 Parallel Workflow Execution

The desired three-tier architecture can be set up with available software components, namely, Taverna and UNICORE components. But the highly recommended security standards cannot be met with this standard. Security issues arrive at Taverna Server, in the second tier. At this tier, the workflow execution takes place (refer to Fig. 10.5). The workflow at some cases has some grid activity, which has to be submitted to the infrastructure. While this submission requires the user certificate, which is not available in Taverna Server, as it is being distributed to it.

Now to equip this situation, the new mechanism based on SAML assertion was investigated in Taverna Workbench and server. SAML is based on trust delegation utilizing user certificate and Taverna Server certificate on the client.

During the requisition to SO, the SO fetches the distinguished name (DN) of the server, and it is provided to Taverna Workbench, which is the client side. The Workbench extracts and stores the DN along with the user's X.509 certificate. This results in trust delegation. This is used by the server to create SAML assertion by which the servers can submit individual applications on behalf of users of HPC.

10.5.3 Parallel Optimization Use Case

In order to achieve the establishment of architecture and novel security, two out of three activities should be optimized. Sub-workflow instances were executed parallel on client and on computing resources. For more detailed analyses of execution scenarios and use cases, refer to [40].

10.6 Optimization Techniques for Workflow Parameters

Generic and automated optimization framework was introduced to fulfill the second requirement to access the scientific workflow optimization. This framework offers API for extension of different optimization levels. The nonlinear problem is approached via proposing a best plug-in, which applies meta-heuristic optimization algorithm. The plug-in actually implements parameter optimization. Therefore the developed framework is extendable by any combination of optimized algorithm and level. Life science workflows combine various data and configuration inputs and contain several different applications. Each component has number of parameters, on which final result critically depends on. In addition to these, the data and configuration inputs also have different influences on the final result. Currently only one application and their specific scope have been optimized, and the configuration parameters are calibrated using default values/trial and error/parameter sweeps. In order to optimize, the measure of the workflow result is required. This measure will be evaluated regarding the numerical value, which is called as objective function. The objective function is calculated by a standard performance measure or simulation software.

It is identified as deterministic, stochastic, and heuristic algorithms can be used to solve the nonlinear optimization problems. Genetic algorithms and other meta-heuristic algorithms need a significant lower number of explored solutions in order to give optimal solutions. Since workflow takes huge time for execution, the distributed and parallel execution environments like evolutionary algorithms are used.

Genetic algorithm: Each parameter is represented via one gene and hence on chrome has one parameter set in it. The constraints give values for genes, and the population represents one generation of several different workflow setups.

Particle Swarm Optimization: Each parameter spans one dimension of search space, and one particle represents one parameter set. The constraints define the range of their respective dimensions.

Ant colony optimization: Each set of parameter is defined by an ant and each visited node represents one parameter. The constraints define range to pheromone intensities.

To obtain best decision in reasonable time, parallel execution must be practiced, and hence the evolutionary algorithms have been emerged as best candidate for it.

10.6.1 Genetic Algorithm

Most popular heuristic optimization methods have been used for more than 36 years to solve complex scientific optimization and search problems. GA has also been used to provide protein structure in real time and calculate multiple sequence alignment and parameter estimation in kinetic models. GA doesn't require any derivations and hence can be easily encoded in both numerical and nonnumerical problems. GA mimics the mechanism of natural evolution by applying recombination, mutations, and solution to a population of parameter vectors. Each of those generations consists of population of individuals which represent a point, which is the possible solution in the search space. Evolution of the population of potential solution is guided by a problem-specific objective function, which is called as fitness function. Each individual is evaluated by this fitness function. The fitness function determines the probability of a solution that has to be inherited to the next generation. The new population is created via selecting the fittest parent of the prior population. The selection is done via applying crossover or mutations. This search space process combines exploration and hill climbing and allows the algorithm to sample many local minima. GA can terminate the process at fixed iterations or at some fixed fitness-based threshold.

To optimize the parameters of the scientific workflows by GA, a mapping between the genetic entities is required. In this, the real-coded GA is selected due to its lower computational complexity compared with the binary-coded GA. Workflow parameters were straightly encoded as genes on the chromosomes. Thus simply, each chromosome encodes set of input parameters and represents a particular combination of input values. GA libraries in different languages are available. For this work, the Java-based library JGAP [41] is selected. In addition to this, library ECJ [42] and Watchmaker framework [43] were also taken into account.

JGAP's application can be described as follows: open-source GA library developed by various freelancing developers. It offers general mechanism for genetic evolution which can be adapted to the particular optimization problem. Parallel execution mechanism and breeder mechanism are best suited for workflow optimization. For supporting the constrained parameter optimization, we make use of a special genotype called Maphene. They are used to define a fixed set, from which breeder can select only one of parameter values. Novel mechanism was inverted to add functions to numerical genes for mapping gene values to the parameter values as given below where y represents the parameter value and x represents the gene value.

$$y = f(x)$$

Main changes done in JGAP are the central breeder method. Refined breeder acts as given below:

- Generated population of sub-workflows is first executed, and the calculated ones are translated into fitness values, in order to determine the fittest chromosomes.

- Then a new population is generated by applying prior chosen natural selection mechanism.
- Crossover or mutations are applied to the parents, and they are added to the population. For crossover, new blend crossover (BLX) mechanism was implemented, and it is extended in JGAP. While mutation is achieved by the uniform mutation operator.
- In order to meet the predefined population size, randomly created chromosomes are added to the population.
- JGAP library suspends and waits until all the sub-workflows have been executed and fitness values have been extracted and returned.

10.7 Parameter Optimization Plug-In

In order to allow automated parameter optimization of workflow within Taverna, an approach to accomplish GA-based optimization plug-in was introduced. By utilizing the generic framework, developers can ignore all the Taverna-specific internals like security, execution and data handling on grid, and GUI. For architecture and development, different requirements have to be taken into account like:

- Provide a panel for optimization configuration.
- Provide methods to start the optimization process.
- Provide methods to return optimization samples.
- Provide methods for receiving a result.
- Provide termination criteria and status methods.
- Offer a method to request the final result.

The parameter optimization plug-in is integrated using the predefined interface methods, into framework API (refer to Fig.10.6). By employing genetic framework, developers don't have to deal with Taverna-specific internals like execution/data handling. All accessible parameters of the sub-workflow are included to parameter optimization process. To allow scientists to modify the parameter sampling space, they developed parameter optimization plug-in that integrates a new pane into optimization perspective of Taverna Workbench. In addition to these, a type-dependent set of properties can also be specified for each parameter. This set of parameters is the lower and upper bounds of the mathematical function stated before. The process of optimization is shown as a control flow diagram:

- User initiates the optimization via selecting sub-workflow defining parameters and constraints giving the input.
- During the optimization run, the optimization framework invokes the parameter plug-in, and it forwards the user-selected sub-workflow.
- Sampling the parameter space for new population formation in JGAP is guided by the constraints, dependencies, and other details that are given by the user.
- The resultant parameter set is returned back to the plug-in.

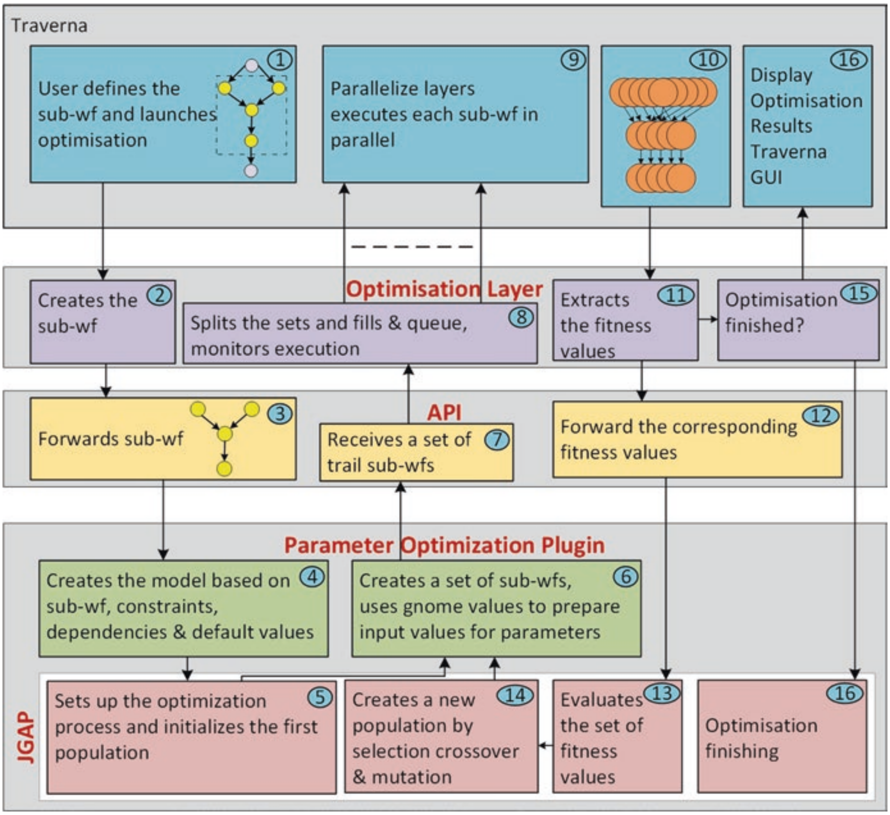

Traverna
User defines the sub-wf and launches optimisation
1
Parallelize layers executes each sub-wf in parallel
9
10
Display Optimisation Results Traverna GUI
16
Optimisation Layer
Creates the sub-wf
2
Splits the sets and fills & queue, monitors execution
8
Extracts the fitness values
11
Optimisation finished?
15
API
Forwards sub-wf
3
Receives a set of trail sub-wfs
7
Forward the corresponding fitness values
12
Parameter Optimization Plugin
Creates the model based on sub-wf, constraints, dependencies & default values
4
Creates a set of sub-wfs, uses gnome values to prepare input values for parameters
6
JGAP
Sets up the optimization process and initializes the first population
5
Creates a new population by selection crossover & mutation
14
Evaluates the set of fitness values
13
Optimisation finishing
16

Fig. 10.6 Control flow of the new optimization framework

- Optimization plug-in initiates the corresponding set of Taverna sub-workflow.
- Now the optimization framework manages parallel execution of instantiated sub-workflows and forwards the results back to the parameter plug-in.
- Now the parameter plug-in extracts the fitness values and forwards them to JGAP library, which evaluates the relative parameter set performance.
- After that the new generation is created and the evolution continues till the optimal result is reached or termination criteria are attained.
- Final result is given back to the user which can be stored for future references.

10.8 Validating Parameter Optimization Plug-In

To validate the optimization plug-in, several real-world use cases from life sciences must be experimentally tested with reasonable execution environment and find good values for crossover and mutation rate. The test runs are especially for scientific

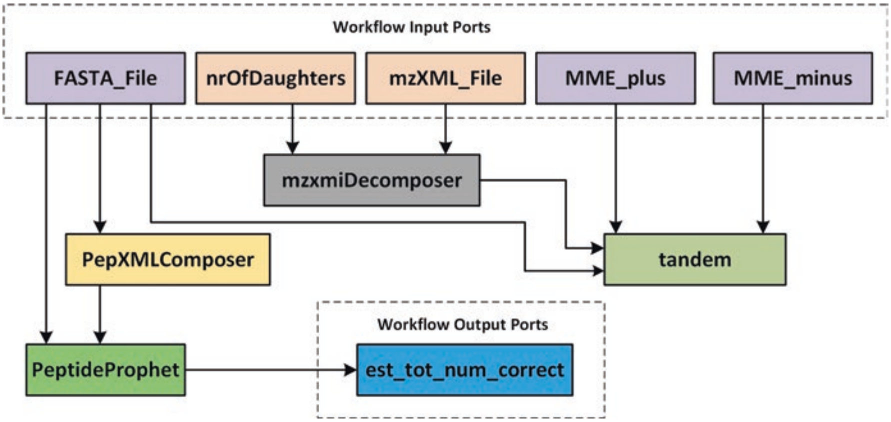


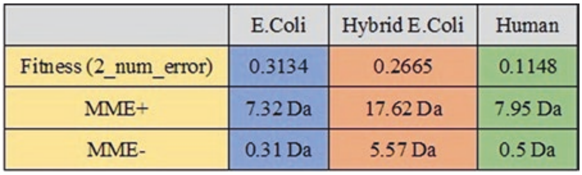
Fig. 10.7 Proteomics workflow using new optimization method

workflow optimization. The benchmarks for these works are tested via Rosenbrock function and Goldstein and price function. These experiments showed good results when the mutation rate and crossover rate are maintained at 0.1 and 0.8 values.

10.8.1 Proteomics Workflow

The primary objective in fundamental analyses of proteins is to gain details about their structure and function. One goal is to determine the composition of a simple via tandem mass spectrometry. Peptide-spectrum matching is one of the most ubiquitous components of data processing workflows in mass spectrometry-based proteomics (refer to Fig. 10.7). A large number of algorithms are embedded in free/commercial software packages to address this issue. However, the use of algorithms requires selection of data-dependent parameters. This selection of parameters depends on several factors, and because of this, their number of returned identified peptides is varied by an order of magnitude which tell us that the final optimal parameters for peptide identification are nontrivial problem. Well, the selected algorithms and parameters require significant knowledge about the data and algorithms. Hence the selection of the algorithms and parameters is decided by the user experience, knowledge level, instrument used, and the quality of the data being used. Because of these, the users will mostly use default or estimated optimal parameters.

The simplified version of optimization workflow is given in figure. It matches the fragment weights to the database for identification of peptides and evaluates the findings. Matching process is conducted by tandem, which is executed on distributed computing infrastructure as an application X!Tandem from TPP toolbox. Tandem uses mzXML file as an input. This file contains the weights for the measured spectra. Database file is in FASTA_FILE format. The mass spectrum is only



	E.Coli	Hybrid E.Coli	Human
Fitness (2_num_error)	0.3134	0.2665	0.1148
MME+	7.32 Da	17.62 Da	7.95 Da
MME-	0.31 Da	5.57 Da	0.5 Da

Fig. 10.8 Results of proteomics workflow

compared against those theoretical peptides, whose mass is within a user-defined maximum mass measured error that is represented by the parameters, MEE+ and MEE-. These parameters are tunable by the workflow.

After the identification of spectra, the PeptideProphet is used to estimate the probability of each assigned peptide. For achieving increased performance, tandem is executed in parallel manner.

Fitness: PeptideProphet concludes a value for the identified spectra. Then spectra are divided by the total number of tandem mass spectra, to serve a fitness value for this use case. This fitness value is denoted as 2_num_corr.

Data input: The input is taken from both prokaryotic and eukaryotic cells. For prokaryotic cell, *Escherichia coli* (*E. coli*) is subjected to limited number of modifications and prepared as described by the Mostovenko et al. It is analyzed by ultrahigh resolution TOF (time of flight) and also by ion trap. They are named as *E. coli* and hybrid *E.coli*, respectively. For eukaryotic cell, dendritic cell from humans is used. Uniprot reference proteome data is used for both *E. coli* and humans for peptide identification.

Optimization: Optimized parameters are maximum mass measured error values, MME+ and MME-, and for the sake of the fixed parameters, these are used so mzXML_File, FASTA_File, and nrOfDaughters. User constraints for the parameters are kept as follows: MME+ ∈ [0,0.5] (double), MME- ∈ [0,0.5] (double), and MME+ − MME− | < 0.2.

In the first iteration, *E.coli* is used and the values are set as MME+ = 0.35, MME− = 0.35, 2_num_corr = 0.1596. The expected optima MME values are between 0.3 and 0.4, while the obtained result values are not satisfactory as they gave 2_num_corr = 0.2636 with MME+ = 0.486 and MME− = 0.4672. Hence the second iteration is carried out.

During the second iteration, *E.coli*, hybrid *E.coli*, and dendritic cell from humans are used with optimization parameters as MME+ = 0.5 and MME− = 0.5, fitness value as 0.2841, 0.2274, and 0.1019 for *E.coli*, hybrid *E.coli*, and dendritic cell from humans, respectively. The results of the experiment are given in Fig. 10.8.

The population size is fixed as 20 and the maximum runtime is scheduled to 24 h. The *E. coli* had six generations and a runtime of around 4 h, and the hybrid *E.coli* had five generations and a runtime of around 4.30 h, while the dendritic cell of human being had six generations with runtime of around 10hr. Here the generation represents that after that many generation, they have been terminated as it showed optimized values in that.

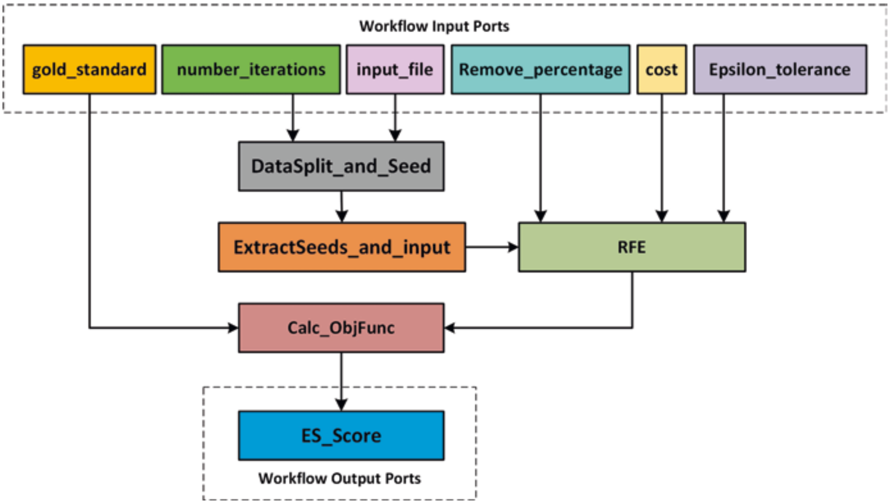
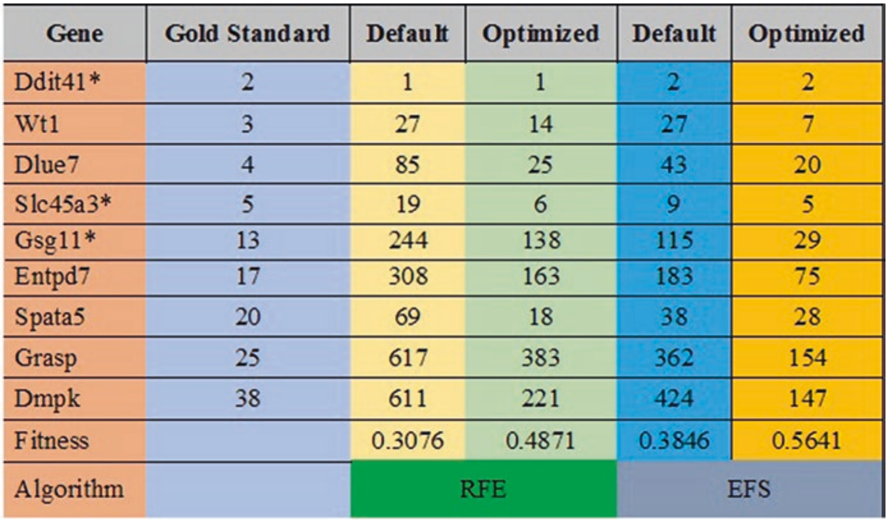


Fig. 10.9 Biomarker identification workflow

10.8.2 Biomarker Identification Workflows

In disease diagnosis and biomedical applications, the important topic is identification of potential biomarkers, which is represented by differentially expressed genes in microarray data. Most of the traditional statistical methods can differentiate normal and differentially expressed genes successfully but with a lesser prediction performances. This results in constant research in this domain. This selection of the relevant gene from a large high-dimensional gene is achieved by the feature selection methods. These feature section methods use machine learning techniques for that purposes. Example for these is recursive feature elimination (RFE) algorithm and ensemble feature selection (EFS). RFE is based on recursive modeling, and it uses absolute value of weights of each dimension as importance of each gene in dataset, while EFS is an ensemble-based concept where multiple feature selections are combined to enhance the robustness. Models are built by subsampling the original dataset for several times. By doing this, the highly ranked genes are found which can be selected as potential biomarkers.

The biomarker identification workflow is shown in Fig. 10.9. The original components are split into several subsampling sets by the RFE and EFS components. RFE performs by executing several instances of SVM, and it does machine learning. The parallel execution of SVM takes place on grid. Nearly 100 instances of SVMs are executed in a parallel fashion. EFS adds another level of sampling to RFE called as bootstrapping. The fitness value is calculated by using calc_ObjFunc, and it is named as ES_Score.



Gene	Gold Standard	Default	Optimized	Default	Optimized
Ddit41*	2	1	1	2	2
Wt1	3	27	14	27	7
Dlue7	4	85	25	43	20
Slc45a3*	5	19	6	9	5
Gsg11*	13	244	138	115	29
Entpd7	17	308	163	183	75
Spata5	20	69	18	38	28
Grasp	25	617	383	362	154
Dmpk	38	611	221	424	147
Fitness		0.3076	0.4871	0.3846	0.5641
Algorithm		RFE		EFS	

Fig. 10.10 Biomarker identification workflow

Fitness function: The function of calc_ObjFunc is to compare the ranked list of genes with the so-called golden standard set. The golden set is produced by using t-test, and it will list the top 40 ranked genes.

Data input: The mouse dataset is taken from the publicly available Gene Expression Omnibus (GEO) database. The Huntington's disease of 24-month-old data is collected.

Optimization: The optimized parameters are taken as remove_percentage, cost, and epsilon_tolerance. The fixed parameters are input_file, number_iterations, gold_standard, svm_type, kernel_type, degree, gamma, coef0, nu, epsilon_loss, cache_size, shrinking, and probability. The user constraints are selected as r move_percentage, cost, epsilon_tolerance, and bootstrapping. The default values for the parameters are given as follows:

- remove_percentage = 0.2
- cost = 1
- epsilon_tolerance = 0.001
- bootstrapping = 40
- RFE_fitness = 0.3076
- EFS_fitness = 0.3846

The size of population is set as 20, and it is terminated at 16th generation, and its runtime duration is 3.20 min. The optimization results are shown in Fig. 10.10.

10.9 Conclusion

Since trial and error parameter sweeps are impractical and inefficient for complex life science workflows, hence, optimization of complex scientific workflow is crucial in novel research area and in silico experiments. The things that are introduced in this approach are optimization phase, plug-ins for achieving the levels of optimization, and automated and generic optimization framework, and that framework was integrated to Taverna. The framework was developed to offer scientists a compensation tool achieving optimized scientific results. The framework and Taverna integration helped developers so that they don't have to deal with Taverna's GUI implementations, specifications, and model execution. Also the issues of security are solved via trust delegation in client and SAML assertion in Taverna Server.

References

1. Sobieszczanski-Sobieski J, Haftka RT (1997) Multidisciplinary aerospace design optimization: survey of recent developments. Struct Optim 14(1):1–23
2. Nicolaou CA, Brown N (2013) Multi-objective optimization methods in drug design. Drug Discov Today Technol 10(3):e427–e435
3. Nocedal J, Wright SJ (2006) Numerical optimization, 2nd edn. Springer, New York
4. Marcelo J. Colaço, George S. Dulikravich (2009) A survey of basic deterministic, Heuristic and hybrid methods for single objective optimization and response surface generation. Thermal Measurements and Inverse Techniques, pp 355–405
5. Debye P (1909) Näherungsformeln für die Zylinderfunktionen für große Werte des Arguments und unbeschränkt veränderliche Werte des Index. Mathematische Annalen 67(4):535–558
6. Holtzer AM (1954) The collected papers of Peter J. W. Debye. Journal of Polymer Science 13(72):548–548
7. Land AH, Doig AG (1960) An automatic method of solving discrete programming problems. Econometrica: Journal of the Econometric Society 28(3):497–520
8. Scott Kirkpatrick C, Gelatt D, Vecchi MP (1983) Optimization by simulated annealing. Science 220(4598):671–680
9. Pham DT, Karaboga D (2000) Intelligent optimisation techniques: genetic algorithms, Tabu search, simulated annealing and neural networks. Springer, London
10. Siarry P, Michalewicz Z (2008) Advances in Metaheuristics for hard optimization. Springer, Berlin
11. Gendreau M, Potvin J-Y (eds) (2010) Handbook of Metaheuristics, 2nd edn. Springer, Boston
12. Jakob Vesterstrøm (2005) Heuristic algorithms in bioinformatics. PhD thesis, Bioinformatics Research Center, Department of Computer Science, Faculty of Science, University of Aarhus
13. Goldberg DE, Holland JH (1988) Genetic algorithms and machine learning. Mach Learn 3(2):95–99
14. Holland JH (1992) Genetic algorithms. Sci Am 267(1):66–72
15. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: IEEE international conference on neural networks, vol 4, pp 1942–1948
16. Engelbrecht AP (2005) Fundamentals of computational swarm intelligence, vol 1. Wiley, Chichester
17. Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195

18. Zhou A, Qu B-Y, Li H, Zhao S-Z, Suganthan PN, Zhang Q (2011) Multiobjective evolutionary algorithms: a survey of the state of the art. Swarm Evol Comput 1(1):32–49
19. Deb K (2001) Multi-objective optimization using evolutionary algorithm*s*. Wiley, Chichester
20. Jean-Charles Boisson, Laetitia Jourdan, El-Ghazali Talbi, Dragos Horvath (2008) Parallel multi-objective algorithms for the molecular docking problem. In: IEEE symposium on computational intelligence in bioinformatics and computational biology, IEEE, pp 187–194
21. Namasivayam V, Bajorath J (2012) Multiobjective particle Swarm optimization: automated identification of structure–activity relationship-informative compounds with favorable physicochemical property distributions. J Chem Inf Model 52(11):2848–2855
22. Poladian L, Jermiin LS (2006) Multi-objective evolutionary algorithms and phylogenetic inference with multiple data sets. Soft Comput 10(4):359–368
23. Hisao Ishibuchi, Tadahiko Murata (1996) Multi-objective genetic local search algorithm In: Proceedings of IEEE international conference on evolutionary computation, IEEE, pp 119–124
24. Moscato P (1999) Memetic algorithms: a short introduction. In: Corne D, Dorigo M, Glover F, Dasgupta D, Moscato P, Poli R, Price KV (eds) New ideas in optimization. McGraw-Hill Ltd, Maidenhead, pp 219–234
25. Blum C, Puchinger J, Raidl GR, Roli A (2011) Hybrid metaheuristics in combinatorial optimization: a survey. Appl Soft Comput 11(6):4135–4151
26. Altschul SF, Gish W, Miller W, Myers EW, Lipman DJ (1990) Basic local alignment search tool. J Mol Biol 215(3):403–410
27. Altschul SF, Madden TL, Schäffer AA, Zhang J, Zhang Z, Miller W, Lipman DJ (1997) Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. Nucleic Acids Res 25(17):3389–3402
28. Wieczorek M, Hoheisel A, Prodan R (2009) Towards a general model of the multi-criteria workflow scheduling on the grid. Futur Gener Comput Syst 25(3):237–256
29. Yu J, Kirley M, Buyya R (2007) Multi-objective planning for workflow execution on grids. In: Proceedings of the 8th IEEE/ACM international conference on grid computing, IEEE, pp 10–17
30. Kumar VS, Kurc T, Ratnakar V, Kim J, Mehta G, Vahi K, Nelson YL, Sadayappan P, Deelman E, Gil Y, Hall M, Saltz J (2010) Parameterized specification, configuration and execution of data-intensive scientific workflows. Clust Comput 13(3):315–333
31. Chen W-N, Zhang J (2009) An Ant colony optimization approach to a grid workflow scheduling problem with various QoS requirements. IEEE Trans Syst Man Cybernetics C Appl Rev 39(1):29–43
32. Yu J, Buyya R, Ramamohanarao K (2008) Workflow scheduling algorithms for grid computing. In: Metaheuristics for scheduling in distributed computing environments, vol 146, studies in computational intelligence. Springer, Berlin, pp 173–214
33. Prodan R, Fahringer T (2005) Dynamic scheduling of scientific workflow applications on the grid: a case study. In: Proceedings of the 2005 ACM symposium on Applied computing, ACM, pp 687–694
34. Abramson D, Bethwaite B, Enticott C, Garic S, Peachey T, Michailova A, Amirriazi S (2010) Embedding optimization in computational science workflows. J Comput Sci 1(1):41–47
35. Ludäscher B, Altintas I, Berkley C, Higgins D, Jaeger E, Jones M, Lee EA, Tao J, Zhao Y (2006) Scientific workflow management and the Kepler system. Concurr Comput Pract Exp 18(10):1039–1065
36. Crick T, Dunning P, Kim H, Padget J (2009) Engineering design optimization using services and workflows. Philos Trans R Soc A Math Phys Eng Sci. 367(1898):2741–2751
37. Missier P, Soiland-Reyes S, Owen S, Tan W, Nenadic A, Dunlop I, Williams A, Oinn T, Goble C (2010) Taverna, reloaded. In: Proceedings of the 22nd international conference on Scientific and statistical database management, Springer, pp 471–481
38. Barga R, Gannon D (2007) Business versus scientific workflow, Workflows for e-Science. Springer, London, pp 9–16

39. Görlach K, Sonntag M, Karastoyanova D, Leymann F, Reiter M (2011) Conventional Workflow Technology for Scientific Simulation, Guide to e-Science, Computer communications and networks. Springer, London, pp 323–352
40. Holl S, Zimmermann O, Demuth B, Schuller B, Hofmann-Apitius M (2012) Secure multilevel parallel execution of scientific workflows on HPC grid resources by combining Taverna and UNICORE Services. In: Proceedings of UNICORE Summit 2012, Schriften des Forschungszentrums Jülich, IAS Series 15, Forschungszentrum Jülich, 2012, pp 27–34
41. Klaus Meffert, Neil Rotstan, Chris Knowles, Ugo Sangiorgi JGAP – Java Genetic Algorithms and genetic programming Package, 2011. URL: http://jgap.sf.net. Accessed 13 May 2013
42. White DR (2012) Software review: the ECJ toolkit. Genet Program Evolvable Mach 13(1):65–67
43. Dyer DW Watchmaker framework for evolutionary computation, 2010. URL: http://watchmaker.uncommons.org/. Accessed 13 May 2013

Chapter 11
The Hybrid IT, the Characteristics and Capabilities

Abstract With the faster adoption of the cloud idea across industry verticals with all the elegance and the enthusiasm, the traditional IT is bound to enlarge its prospects and potentials. Thatis, the IT capabilities and capacities are being enhanced with the seamless and spontaneous association with the cloud paradigm in order to meet up fast-emerging and evolving business requirements. This is a kind of new IT getting enormous attention and garnering a lot of attraction among business executives and IT professionals lately. The systematic amalgamation of the cloud concepts with the time-tested and trusted enterprise IT environment is to deliver a bevy of significant advantages for business houses in the days ahead. This model of next-generation computing through the cognitive and collective leverage of enterprise and cloud IT environments is being touted as the hybrid IT. There are a variety of technologies and tools expressly enabling the faster realization of hybrid era. This chapter is specially crafted for digging deep and describing the various implications of the hybrid IT.

11.1 Introduction

The cloud paradigm is definitely journeying in the right direction toward its ordained destination (the one-stop IT solution for all kinds of institutions, innovators, and individuals). The various stakeholders are playing their roles and responsibilities with all the alacrity and astuteness to smoothen the cloud route. Resultantly there are a number of noteworthy innovations and transformations in the cloud space, and they are being consciously verified and validated by corporates in order to avail them with confidence. There are IT product vendors, service organizations, independent software vendors, research labs, and academic institutions closely and collaboratively working to make the cloud idea decisively penetrative and deftly pervasive.

In this chapter, we are to describe the various unique capabilities of hybrid clouds and how the feature-rich and state-of-the-art IBM hybrid cloud offering differentiates well against the various hybrid cloud service providers in the competition-filled market. Hybrid cloud ensures fast and frictionless access for cloud infrastructures, platforms, software, and data along with the much-touted bulletproof governance

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6_11

and IT service management (ITSM) features. However, establishing and sustaining hybrid cloud facility is beset with a number of challenges including maintaining consistent configuration and developer experiences across geographically distributed cloud environments. The other prominent issues include workload modernization and migration, IT cost management, and monitoring. There are multiple automated solutions for eliminating the known and unknown complexities of hybrid cloud monitoring, measurement, and management. This report elucidates the right and relevant things on various capabilities of different hybrid cloud service providers and the tools they are using to accelerating and augmenting hybrid cloud processes. This chapter focuses on the following value-adding and decision-enabling parameters:

1. Hybrid cloud-enablement capabilities
2. Cloud migration
3. Cloud bursting, connectivity, and integration features
4. Cloud orchestration capabilities
5. The realization of cloud-enabled and native applications
6. Cloud brokerage services for multi-cloud environments
7. Software configuration, deployment, and management
8. Hybrid cloud management capabilities (through third-party as well as homegrown tools)
9. Application performance management (APM)
10. Service integration and management
11. Distributed deployment and centralized management (a single pane of glass to manage resources and applications across hybrid clouds)

Let us see how the various service providers cope up with these identified parameters, and this work got initiated in order to give the right and relevant insights such as which provider stands where in the challenging yet promising domain of hybrid cloud.

11.2 Demystifying the Hybrid Cloud Paradigm

Let us start with a brief of the raging hybrid cloud concept. The cloud paradigm is definitely recognized as the most disruptive one in the IT space in the recent past. The cloud idea has brought in a series of well-intended and widely noticed business transformations. It all started with the widespread establishment and sustenance of public clouds that are typically centralized, consolidated, virtualized, shared, automated, managed, and secured data centers and server farms. This trend clearly indicated and illustrated the overwhelming idea of IT industrialization. The era of commoditization to make IT the fifth social utility has also flourished with the faster maturity and stability of cloud concepts. That is, public clouds are made out of a large number of commodity server machines, storage appliances, and networking devices to make IT cost-effective, elastic, and energy efficient. The volume-based and value-added IT service delivery methods through geographically distributed and standard-compliant cloud environments have made the cloud concept commercially viable and enviable. The much-discussed compartmentalization technologies represented by virtualization and containerization are playing a vital role toward the unprecedented success of the cloud paradigm. The IT agility, adaptability, and affordability succulently imparted by the public cloud paradigm have led to a bevy of newer business models and the much-anticipated IT operational efficiency.

The astounding success of public clouds has in turn invigorated worldwide enterprises to explore the process of setting up of enterprise-grade private clouds. The existing on-premise data centers are being systematically cloud-enabled to establish a seamless integration with publicly available clouds. On the other hand, the proven and promising cloud technologies and tools are being applied to establish local cloud environments from the ground up. There are several crucial requirements for having on-premise private clouds for certain industry verticals. That is, to ensure an exemplary control, visibility, and security of enterprise IT, private clouds are being insisted. The business-critical requirements such as high performance, availability, and reliability also have forced worldwide enterprises to have their own clouds on their premises.

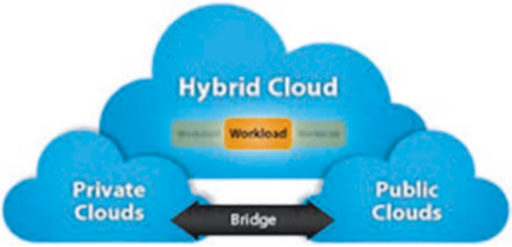
The aspect of convergence has become the new normal in the IT space. Having understood that there are elegant and extra advantages with the combination of private and public clouds, there is a renewed interest and rush by businesses to embrace the hybrid version of clouds. There are certain scenarios and benefits with hybrid clouds. Cloud service providers are equally keen in providing hybrid capabilities to their clients and consumers.

Hybrid cloud is a combination of on-premise and public cloud services to bring forth an exemplary value for businesses. The hybrid IT is the most influential and important moment for worldwide businesses in order to guarantee enhanced value for their customers, consumers, partners, employees, etc. The hybrid environment provides customers with all the flexibility and extensibility to select and consume the most appropriate service offering for specific workloads based on several critical factors such as cost, security, and performance. For example, a customer may choose a public cloud service to test and develop a new application and then move that workload to a private cloud or traditional IT environment when the application becomes operational. Enterprises that need to support a variety of workloads can leverage the flexibility of a hybrid cloud approach to ensure they have the ability to scale up and scale down as needed. The emerging hybrid capability through a bevy of automated tools is a blessing in disguise for worldwide enterprises to meet up the fast-evolving business sentiments.

11.3 The Key Drivers for Hybrid Clouds

Hybrid cloud is a kind of converged cloud computing environment which uses a mix of on-premise private cloud and public cloud services with seamless interaction and orchestration between the participating platforms. While some of the organizations are looking to put selective IT functions onto a public cloud, they still prefer keeping the higher-risk or more bespoke functions in a private/on-premise environment. Sometimes the best infrastructure for an application requires both cloud and dedicated environments, that is, being touted as the prime reason for having hybrid clouds.

- *Public cloud* for cost-effective scalability and ideal for heavy or unpredictable traffic
- *Private cloud* for complete control and security aspects
- *Dedicated servers* for super-fast performance and reliability



A hybrid cloud configuration offers the following features:

- *Flexibility* – The availability of both scalable, cost-effective public resource and secure resource can provide better opportunities for organizations to explore different operational avenues.
- *Cost efficiencies* – Public clouds are likely to offer significant economies of scale (such as centralized management) and greater cost efficiencies than the private cloud. Hybrid cloud model, therefore, allows organizations to unlock these savings for as many business functions as possible and still keeping sensitive operations secure.
- *Security* – The private cloud feature in the hybrid cloud model not only provides the security where it is required for sensitive operations but can also meet regulatory requirements for data handling and storage where it is applicable.
- *Scalability* – Although private clouds offer a certain level of scalability based on the configurations (whether they are hosted externally or internally), public cloud services do offer scalability with fewer boundaries as resources are pulled from the larger cloud infrastructure. By moving as many nonsensitive functions as possible to the public cloud infrastructure, this would allow organizations to benefit from public cloud scalability even as reducing the demands on a private cloud.

Hybrid clouds succulently empower enterprises to innovate rapidly while fulfilling the enterprise-grade performance, resiliency, and security requirements. Hybrid cloud ensures the widely articulated enterprise IT needs by immaculately combining the control and reliability of private cloud with the scalability, consumability, and cost efficiency of public clouds. Leveraging hybrid cloud environments enables you to run every workload in its optimal place at an optimal cost.

Integration of Applications, Data, and Services A hybrid cloud creates the transparency needed to see and connect data and applications across infrastructures. For example, a hybrid cloud approach can foster integration between internal systems of record, often housed on traditional IT or on a private cloud, and more outward-facing systems of engagement, which are increasingly hosted on a public cloud.

Composition and Management of Workloads An agile and competitive business is increasingly a composable business. In any composable business, all sorts of processes, applications, services, and data become building blocks. These blocks are quickly and easily found, bound, and assembled and reassembled in the cloud to find new ways to rapidly innovate and engage with customers. Distributed and different cloud environments combine well to realize composable businesses. A hybrid cloud enhances developer productivity so applications can be integrated, composed, and delivered.

Portability of Data and Applications In a hybrid environment, developers can rapidly connect and compose data and services for the enterprise, the Web, and mobile applications, allowing organizations to act fast. Perhaps you need to make an application available in a new country or move from a development and test environment to production or move from primary capacity to scale-out capacity

Flexibility with Speed Hybrid clouds offer the broadest choice of platforms and infrastructures on which to build and deploy a range of applications at the speed required for business needs.

Value-Driven with Variety Ubiquitous access to data sources for applications, a growing software market store, and integrated platforms across on-premise and off-premise clouds.

Reliability with Resiliency Unbreakable and impenetrable data security and application resiliency are the distinct hallmarks of hybrid clouds.

Cost Optimization The choice of cloud environments for efficiently and affordably running application workloads is being facilitated through hybrid clouds.

Enhanced Utilization By leveraging the already invested and installed IT infrastructures, the IT costs could be kept low. Underutilized and unutilized infrastructural resources can be used to the highest value.

Hybrid Cloud Use Cases It is possible to achieve higher levels of control, reliability, availability, elasticity, quality, and performance in hybrid cloud environments. Customers have acknowledged the following five key use cases for a hybrid cloud implementation.

Development and Testing Hybrid cloud provides businesses with the required flexibility to gain the needed capacity for limited time periods without making capital investments for additional IT infrastructures.

Extending Existing Applications With a hybrid cloud, businesses can extend current standard applications to the cloud to meet the needs of rapid growth or free up on-site resources for more business-critical projects.

Disaster Recovery Every organization fears an outage, or outright loss, of business-critical information. While on-site disaster recovery solutions can be expensive, preventing businesses from adopting the protection plans they need, a hybrid cloud can offer an affordable disaster recovery solution with flexible commitments, capacity, and cost.

Web and Mobile Applications Hybrid cloud is ideal for cloud-native and mobile applications that are data-intensive and tend to need the elasticity to scale with sudden or unpredictable traffic spikes. With a hybrid cloud, organizations can keep sensitive data on-site and maintain existing IT policies to meet the application's security and compliance requirements.

Development Operations As developers and operations teams work closer together to increase the rate of delivery and quality of deployed software, a hybrid cloud allows them to not only blur the lines between the roles but between Dev/Test and production and between on-site and off-site placement of workloads.

Capacity Expansion Quickly addresses resource constraints by bursting workloads into VMware on IBM cloud.

Data center Consolidation Consolidate legacy infrastructures onto an automated and centrally managed software-defined data center.

The adoption of hybrid clouds is being driven due to several parameters as articulated above. The mandate toward highly optimized and organized cloud environments for enabling smarter organizations is the key force for hybrid clouds. The heightened stability and surging popularity of hybrid clouds ultimately lead to multi-cloud environments.

11.4 The Hybrid Cloud Challenges

Hybrid cloud facilitates to run different applications in the best of the environments to reap the required advantages such as the speed, scale, throughput, visibility, control, etc. There are competent solutions and services being offered by different providers in the cloud space in order to accelerate the hybrid cloud setup and the sustenance. There are cloud infrastructure service providers and cloud-managed service providers too in plenty. There is a plethora of open-source as well as commercial-grade solutions and toolsets. Service providers have formulated and enabled frameworks toward risk-free and sustainable hybrid clouds.

However, there are challenges too. Especially the prickling challenge lies in establishing a high-performing hybrid environment to appropriately and accurately manage and monitor all of its different components. Most of the current infrastructure management and monitoring tools were initially built to manage a single environment only. These point tools are incapable of managing distributed yet connected environments together. These tools lack the much-needed visibility and controllability into different environments; thereby the activities such as workload migration among the integrated clouds are not happening smoothly. Further on, the application performance management (APM) across the participating clouds is also not an easy affair.

Thus, there is an insistence for integrated cloud management platform (CMP) in order to leverage the fast-evolving hybrid concept to the fullest extent to give utmost flexibility to businesses.

The Distinct Capabilities of Hybrid Clouds

We are slowly yet steadily heading toward the digital era. The digitization-enablement technologies and tools are bringing a number of rapid and radical changes on how we collaborate, correlate, corroborate, and work. IT professionals are under immense pressure to meticulously capitalize these innovations and help their organization to move with agility and alacrity than today. The mantra of "more with less" has induced a large number of organizations to accurately strategize and implement a private cloud due to the huge advantages being offered by public clouds. But with the broad range of cloud platforms, along with the explosion of new infrastructure and application services, having a private cloud environment is no longer sufficient and efficient. The clear-cut answer is the realization of hybrid clouds. The following questions facilitate to understand how hybrid cloud comes handy in steering businesses in the right direction:

- Are workloads moved from private to public environments?
- Do you develop the Web or mobile application in one cloud platform but run it in a different cloud?
- Do your developers want to use multiple public platforms for their projects?

The following are the widely articulated and accepted features of hybrid cloud service providers:

1. *Cloud bursting* – After all, the promise of running a performant and efficient private data center and leveraging public cloud providers for the occasional hybrid cloud bursting is the way forward. On the private side, you can maintain a complete control over privacy, security, and performance for your workloads, and on the public side, you can have "infinite" capacity for those occasional workloads. Imagine a Ruby application that is being used for an online store and the transactional volume is increasing due to a sale, a new promotion, or Cyber Monday. The cloud bursting module will recognize the increased load and recommend the right action to address the issue, effectively answering *when* it's time to burst.

It will also give the recommendation of *where* to clone the instance. So it is all about deciding *what* workload to burst and *where* to burst it including specific placement for computing and storage in a public cloud environment. That is, the module enables to extend your private cloud to:

- Burst load into the cloud when demand increases and cannot be met with local resources
- Maintain control on workload performance and resource utilization to decide when to move back when possible
- Allow application to auto-scale and clone into the cloud
- Load balances across private and public clouds

The module continuously analyzes the hybrid cloud resources and IT stack taking into account multidimensions of continuously fluctuating trade-offs: trade-offs between QoS vs. budget and costs; between workload demand and infrastructure supply; between application performance and infrastructure utilization; between computing, storage, and network; etc.

2. *VM migration support* – As hybrid clouds increase in popularity, it is important for organizations to be able to move virtual machines (VMs) from an on-premise hypervisor to the public cloud and to bring those workloads back to the house if necessary.
3. *Custom image support* – Cloud providers generally allow VMs to be built from predefined images, but these generic OS images don't always meet an organization's needs. As such, a cloud provider should allow custom virtual machine images to be created and used.
4. *Image library* – Although many organizations try to minimize the number of server operating systems they use, heterogeneous environments are becoming much more common, especially in the cloud. A good cloud provider should offer a variety of server OS choices.
5. *Auto-scaling* – Workloads do not typically experience linear demand; instead, demand increases and decreases over time. Ideally, a cloud provider should allow workloads to automatically scale up or down in response to current demand.
6. *Network connectivity* – Network connectivity is another important consideration when choosing a cloud provider. There should be a way to connect your on-premise network to your cloud network, and the provider should offer various connectivity features.
7. *Storage choices* – Storage needs vary depending on workloads. Some workloads can use commodity storage without any issues, while others require high-performance storage. As such, a cloud provider should offer a variety of storage options.
8. *Regional support* – Sometimes a business or regulatory requirements mandate hosting resources in a specific geographic region. That being the case, a cloud provider should ideally give its customers a choice of where VMs will be hosted.

9. *Data backup, archival, and protection* – Cloud environments are turning out to be an excellent mechanism for business continuity and resiliency.
10. *Identity and access management (IAM)* – Authentication, authorization, and audibility are the key requirements for ensuring cloud security and privacy. There are several additional security-ensuring mechanisms for impenetrable and unbreakable security.
11. *Disaster and data recovery* – Whether their applications are running in on-premise private clouds, hosted private clouds, or public clouds, enterprises are increasingly seeing the value of using the public cloud for disaster recovery. One of the major challenges of traditional disaster recovery architectures that span multiple data centers has been the cost of provisioning duplicate infrastructure that is rarely used. By using pay-as-you-go public cloud as the disaster recovery environment, IT teams can now deliver DR solutions at a much lower cost.
12. *Service integration and management* – IT service management (ITSM) has been an important requirement for different IT service providers, data center operators, and cloud centers for exemplary service fulfillment. As clouds emerge as the one-stop IT solution, the aspect of service management and integration garners extreme importance in order to fulfill the agreed SLAs between cloud service consumers and providers. The various nonfunctional attributes need to be guaranteed by CSPs to their subscribers through highly competitive ITSM solutions.
13. *Monitoring, measurement, and management* – Hybrid cloud management platform is a key ingredient for running hybrid cloud environments successfully, and we have extensively written about the role and responsibility of a viable management platform in a hybrid scenario.
14. *Metering and charge-back* – A charge-back model in the cloud delivers many benefits, including the most obvious:

    - Correlating utilization back to cloud consumers or corporate departments, so that usage can be charged if desired
    - Providing visibility into resource utilization and facilitating capacity planning, forecasting, and budgeting
    - Providing a mechanism for the enterprise IT function to justify and allocate their costs to their stakeholder business units

With the continued evolution of business sentiments and expectations, the capabilities of hybrid clouds are bound to enlarge consistently through the careful addition of powerful tools, processes, and practices.

Hybridization is definitely a unique and useful approach for different scenarios. In the cloud space also, the hybrid capability is all set to penetrate and participate in bringing forth a number of exceptional benefits such as the acceleration of innovations, sharply lessening the time to take fresh products and services to the knowledge-filled market, the means and ways of achieving agile development through continuous integration, deployment, and delivery, and guaranteeing the significant enhancement in resource utilization, for worldwide enterprises.

11.5 Hybrid Cloud Management: The Use Cases and Requirements

Hybrid cloud management solutions are typically automation and orchestration platforms. They automate manual or scripted tasks, and they orchestrate tasks and processes that execute across a number of distributed and disparate cloud environments. Forrester, one of the leading market analysts, has identified the four most common hybrid cloud management use cases today:

- Accelerating hybrid cloud application development and delivery
- Managing and governing the hybrid cloud infrastructure life cycle
- Migrating cloud apps and infrastructure among cloud platforms
- Creating an enterprise cloud brokering function

Businesses are involving hybrid cloud management platforms for solving different requirements. If hybrid cloud challenges are primarily around the cloud infrastructure life cycle, then it is logical to focus on features that help package up infrastructure components as per customers' requirements, present them to your cloud consumers, automate deployment and migration, and monitor consumption and performance. If you are more concerned with accelerating cloud development, look for prepackaged application templates, cloud-agnostic and developer-friendly APIs, integrations with application release automation (ARA) tools, and policy-based automation of application life-cycle events like auto-scaling. If cost management is a top concern, prioritize price benchmarking and cost analytic features. The key capabilities are summarized in the reference architecture model in Fig. 11.1.

11.6 The Growing Ecosystem of Hybrid Cloud Management Solutions

The hybrid cloud management landscape is fast enlarging:

- For many incumbent enterprise systems and technology vendors, hybrid cloud management is just an evolution of existing IT service management (ITSM) suites. These are meticulously extended to manage cloud infrastructure endpoints using the same automation tools that the vendor sells to manage physical and virtualized infrastructure in the data center.
- Enterprise technology vendors that have built private cloud solutions often consider hybrid cloud management to be an extension of private cloud.
- There are smaller software companies that created solutions to manage multiple public cloud platforms first, and they are now refactoring them to manage private clouds so that they can be categorized as hybrid cloud management solutions.

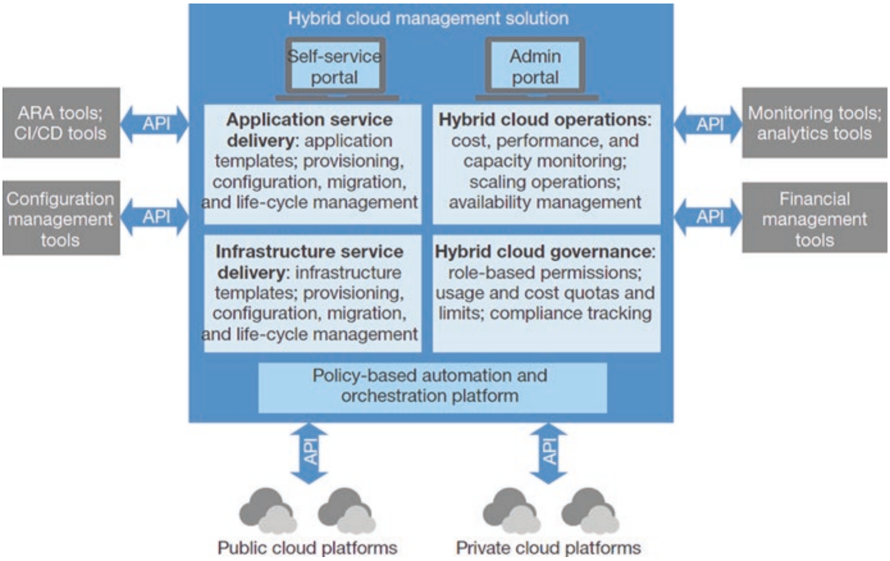

Hybrid cloud management solution
Self-service portal
Admin portal
ARA tools; CI/CD tools
API
Configuration management tools
API
Application service delivery: application templates; provisioning, configuration, migration, and life-cycle management
Hybrid cloud operations: cost, performance, and capacity monitoring; scaling operations; availability management
Infrastructure service delivery: infrastructure templates; provisioning, configuration, migration, and life-cycle management
Hybrid cloud governance: role-based permissions; usage and cost quotas and limits; compliance tracking
API
Monitoring tools; analytics tools
API
Financial management tools
Policy-based automation and orchestration platform
API
API
Public cloud platforms
Private cloud platforms

Fig. 11.1 Hybrid cloud management reference architecture

There are others who built solutions for cloud migration or brokerage and are now extending them to have life-cycle and governance features to be accommodated in the hybrid cloud management solutions:

1. Established management solution providers extend existing management suites. Three of the traditional big four enterprise management vendors (BMC Software, HP, and IBM) offer stand-alone hybrid cloud management solutions, often integrating with and leveraging other tools from the vendor's existing catalog. Microsoft and Oracle have added hybrid cloud management features to System Center and Enterprise Manager, respectively.
2. Enterprise system vendors add hybrid management to private cloud platforms. Cisco Systems, Citrix, Computer Sciences Corp (CSC), Dell, HP, IBM, Microsoft, Oracle, and VMware all offer private cloud suites that may also include hybrid cloud management capabilities. If private cloud is critical to your hybrid cloud strategy, consider whether such an extension of private cloud meets your needs.
3. Hypervisor and OS vendors target technology managers who own infrastructure. Citrix, Microsoft, Red Hat, and VMware offer virtualization platforms and virtual machine (VM)-focused management tools and focus hybrid cloud management solutions on the cloud infrastructure life-cycle use case. CloudBolt and Embotics also feature VM-focused management capabilities.
4. Independent software vendors target cloud-focused developers and DevOps. CliQr, CSC, Dell, DivvyCloud, GigaSpaces, RightScale, and Scalr market their solutions primarily at cloud developers and the DevOps pros who support them. Their solutions are well suited for the cloud application life-cycle use case.

5. Public cloud platform vendors focus on their own clouds in hybrid scenarios. Cisco, IBM, Microsoft, Oracle, Red Hat, and VMware offer public cloud platforms in addition to hybrid cloud management software. Naturally, these vendors encourage the use of their own platforms for hybrid deployments. Pay attention to how strongly the vendor's own platform is favored when evaluating its hybrid cloud management capabilities.
6. Cloud migration vendors add more life-cycle management features. HotLink and RackWare are cloud migration tools with added VM management features that extend to public cloud platforms. They stress the onboarding and disaster recovery use cases for cloud migration. RISC Networks is a cloud migration analysis tool. In addition to these vendors, many other hybrid cloud management vendors in this landscape have migration capabilities.
7. Cloud brokers and brokerage enablers extend beyond cost analytics. AppDirect, Gravitant, Jamcracker, and Ostrato primarily focus on the enterprise cloud brokerage use case. Each of these vendors, however, offers additional capabilities beyond cost brokering and analytics.

11.7 IBM Cloudmatrix-Based Multi-cloud Environments

Due to a large number of connected, clustered, and centralized IT systems, the heterogeneity and multiplicity-induced complexity of cloud centers has risen abnormally. However, a bevy of tool-assisted, standard-compliant, policy- and pattern-centric, and template-driven methods have come handy in moderating the development, management, delivery, and operational complexities of clouds. Precisely speaking, converged, virtualized, automated, shared, and managed cloud environments are the result of a stream of pioneering technologies, techniques, and tools working in concert toward the strategically sound goal of the Intercloud. The results are all there for everyone to see. IT industrialization is seeing the light, the IT is emerging as the fifth social utility, and the digital, insightful, idea and API era are kicking in. Brokerage solutions are being presented and prescribed as the most elementary as well as essential instrument and ingredient for attaining the intended success. In this document, we would like to describe how IBM cloudMatrix, the enterprise-grade cloud brokerage solution, is going to be the real game-changer for the ensuing cloud era.

Undeniably the cloud journey is still at a frenetic pace. The game-changing journey started with server virtualization with the easy availability, the faster maturity, and stability of hypervisors (virtual machine monitors (VMMs)). This phase is thereafter followed by the arrival of powerful tools and engines to automate and accelerate several manual tasks such as virtual machine (VM) monitoring, measurement, management, load balancing, capacity planning, security, and job scheduling. In addition, the unprecedented acceptance and adoption of cloud management platforms such as OpenStack, CloudStack, etc. have made it easy for decisively and declaratively managing various IT infrastructures such as compute machines, storage appliances, networking solutions, OS images, etc. Further on, there are patterns, manifests, and recipe-centric configuration management tools for appropriately configuring, installing, and sustaining business workloads, IT platforms, databases, and middleware. There are also orchestration tools for template-enabled infrastructure provisioning, patching, administration, and governance.

There are ITIL-compliant service management tools for servicing all kinds of cloud infrastructures, resources and applications, operating systems, and application workloads in order to strengthen business continuity, consumability, and customer delight. Thus, the end-to-end life-cycle management of cloud resources and applications is being taken care of through policy-aware and insight-driven integrated tools. The complicated tasks such as workflow/task scheduling for long-running applications, workload optimization through VM consolidation and placement based on varying parameters, and operational analytics are being simplified through pathbreaking algorithms and patentable techniques. There are promising and proven solutions for business process management (BPM), business rule engines, performance engineering, enhancement, etc. to take the cloud enablement to the next level. Now we are heading toward the realization of software-defined cloud environments with not only compute machines but also networking as well as storage solutions are also getting fully virtualized. There are hypervisor solutions for enabling network and storage virtualization. The spectacular advancements in the data analytics and machine/deep learning domains will steadily set up and sustain cognitive clouds in the years ahead.

Thus, the aspect of cloudification is definitely on the right track and direction in order to provide all the originally envisaged business and technical benefits to various stakeholders including cloud service providers, brokers, procurers, auditors, developers, and consumers. Precisely speaking, these widely debated and discoursed technology-driven advancements have collectively resulted in scores of highly optimized and organized hybrid cloud environments.

11.8 The Key Drivers for Cloud Brokerage Solutions and Services

The following are the prominent and dominant drivers for the huge success of the brokerage concept:

1. Transforming to hybrid IT
2. Delivering the ideals of “IT as a service”
3. Planning smooth transition to cloud
4. Empowering self-service IT
5. Incorporating shadow IT
6. Setting and sustaining multi-cloud environments
7. Streamlining multi-cloud governance and control

IBM cloudMatrix is the prime ingredient for enabling hybrid IT – When a data center nears the end of life, an important decision has to be made on how to replace it, and increasingly enterprises are opting to replace their inflexible and complicated data centers with a mix of cloud and physical assets, called hybrid IT. Enterprises are recognizing the need to be more competitive in their dealings, decisions, and deeds. Some of the basic problems they need to solve for are the capital and operational costs, the time to value, the lack of automation, the charge-back accuracy, etc. Hybrid IT helps solve these perpetual problems and increases competitiveness, as long as the right expertise and tools are being leveraged.

1. *Ongoing cost* – The cost of operating, maintaining, and extending application services within the physical data center environments, especially across political and geographic boundaries, would continue to increase.
2. *Speed* – Internal and technology requests for services, on average, took four to six weeks for review and approval, often leading to frustration and a lack of agility for business units.
3. *Lack of automation* – Fulfilling application service requests took too many manual steps, exacerbated by required technology skillsets.
4. *Charge-back accuracy* – Business units were being charged a percentage of IT costs without consideration of usage.
5. *Capital expenditure* – There is a large upfront cost associated with building and deploying new data centers.

The hybrid IT is definitely a long-term, strategic approach and move for any enterprise IT. The hybrid IT typically comprises private cloud, public cloud, and traditional IT. There are some game-changing advantages of hybrid IT. The first and foremost is that it never ask you to rip and replace the current system. Any hybrid IT solution would need to continue to interoperate with the existing service management system and work with ticket management where appropriate. The most crucial tool for realizing painless and risk-free hybrid IT is a highly competitive and comprehensive cloud brokerage solution. A complete cloud brokerage solution would tie planning, consumption, delivery, and management seamlessly across public, private, virtual, hosted, and on- and off-premise solutions. IBM cloudMatrix is widely recognized as the best-in-class cloud brokerage solution. I have given its unique capabilities in comfortably fulfilling the various

IBM cloudMatrix provides the following features:

- A seeded catalog of the industry's leading cloud infrastructure providers, out of the box without the overhead of custom integration.
- A marketplace where consumers can select and compare provider services or add their own IT-approved services for purchasing and provisioning. Consumers can use a common workflow with approval processes that are executed in terms of minutes not weeks.

(continued)

- Reporting and monitoring that includes multi-provider consolidated billing estimates, actuals, and usage projections for accuracy and cost assignment.
- A visual designer that includes sync-and-discover capabilities to pull assets (VMs) into a single, architectural view and management standard.
- Integration with service management and ticketing systems through an API framework.

Creating Hybrid IT Environments As we all know, hybrid cloud is typically a kind of dynamic combination of private and one or two public cloud environments. However, hybrid IT represents a multi-cloud environment by seamlessly integrating geographically distributed and disparate cloud environments in order to gain the strategic advantages of the location, performance, capability, and cost in order to elegantly fulfill the workload requirements and granular business objectives. There are definitely challenges and concerns in the form of multiple locations, application/workload features, governance models, proprietary technologies, etc. to achieve the elusive hybrid IT goal. In the recent past, there came a number of enabling cloud connectors, integrators, adaptors, APIs, and brokers to realize the hybrid IT vision.

Devising and delivering a successful hybrid IT implementation come down to evaluating and managing both traditional and cloud IT, balancing various on-premise and off-premise suppliers, and making dynamic choices about technology on the fly as business requires new capabilities. All of these tasks must be done simultaneously and in tandem to achieve three fundamental aims for success:

1. Providing users and customers with the right service levels for each application and user
2. Optimizing application delivery, streamlining, simplifying, and automating IT operations
3. Enabling service-centric IT that accelerates business responsiveness now and ongoing

But these aims require new approaches. Solutions are no longer wholly contained in the house, on-premises. Technology becomes an ecosystem of providers, resources, and tools. Interactions between old and new IT have to be devised, modeled, tested, implemented, and improved. Many sources of technology have to be managed, integrated, and directed on demand toward business agility. This extended scope requires IT to connect the company with a variety of suppliers and customers – all of which must be juggled effectively to avoid risks or organizational impact. Actually, hybrid IT infrastructure can't be achieved unless IT operates more like a business – managing vendor selection, packaging, pricing, delivery, and billing in a multisourced model. Considering all these correctly, there is an expressed need for enterprise-class, context-aware, highly synchronized, and sophisticated software solution for fulfilling the hybrid IT vision.

Journeying Toward the "IT as a Service (ITaaS)" Days This is definitely service era. The excitement and elegance associated with service-oriented architecture (SOA) have paid well in formulating and firming up the journey toward the days of "everything as a service (XaaS)." The service paradigm is on the heightened growth. The varied tasks such as service conceptualization, concretization, composition, deployment, delivery, management, and enhancement are getting extremely simplified and accelerated through a variety of automated tools. All kinds of IT capabilities are being expressed and exposed as easily identifiable, network-accessible, distinctively interoperable, smartly composable, quickly recoverable, and replaceable services. With such kinds of service enablement, all kinds of closed, monolithic, and inflexible IT infrastructures are being tuned into open, remotely consumable, easily maneuverable, and managed components. With the arrival and acceptance of IT service monitoring, measurement, and management tools, engines, and platforms, the IT assets and applications are being readied for the era of ITaaS.

Further on, microservice architecture (MSA), which is an offshoot of SOA, is gaining a lot of ground these days, and hence the days of "as a service" is bound to see a litany of powerful innovations, transformations, and even a few disruptions. Cloud broker solutions are being recognized as the best fit for ensuring this greatly expressed need of ITaaS.

Embracing the Cloud Idea The raging cloud paradigm is acquiring a lot of attention and attraction because of its direct and decisive contribution toward highly optimized and organized IT environments. However, cloud embarkation journey is beset with innumerable barriers. For jumping on the cloud bandwagon, especially identifying which application workloads give better results in which cloud environments is a tedious and tough job indeed. Herein, a full-fledged cloud broker plays a very vital role in shaping up the cloud strategy and implementation.

Ticking Toward Self-Service IT It is being insisted that IT has to be business and people friendly. For working with IT solutions and availing IT-enabled services, the interfaces have to be very informative, intuitive, and intelligent for giving a simplified and streamlined experience to various users. Automation has to be an inherent and important tenet and trait of cloud offerings. Cloud brokers are being positioned as the principal instrument to have quick and easy servicing of cloud infrastructures, platforms, and applications.

Enabling the Shadow IT Requirements The IT organizations of worldwide enterprises literally struggle to provide the required capabilities with the same level of agility and flexibility as being provided by public clouds. Even their own on-premise private clouds with all the cloud-enabled IT infrastructures do not provide the mandated variety, simplicity, and consumability of public clouds because legacy workflows, manual interpretation and intervention, and business procurement requirements often reduce the accelerated realization. These challenges increasingly drive business users to search for and procure various IT capabilities without involving the core IT team of their organizations. That is, different departments and users within a corporate on their own fulfill their IT requirements from various public clouds. They circumvent the core IT team, and this industry trend is being called as the shadow IT. Users use a shadow IT model because public clouds guarantee on-demand resources, and this, in turn, lays a stimulating foundation for accelerating innovation and improving time to market for newer and premium offerings.

However, the shadow IT is beset with risks and challenges, and there is an intense pressure on IT divisions of business houses to address this in a structured and smart manner. Many IT organizations don't know what cloud services their employees are using. The IT team doesn't know where data resides, whether datasets are safeguarded accordingly, whether data and applications are backed up to support data and disaster recovery, whether the capabilities will scale in line with fast-evolving business sentiments, and what the costs are. Thus, it is becoming mandatory for business behemoths to address this issue of shadow IT by offering a compelling alternative. The traditional IT centers and even private clouds need to be empowered to give all that are being ensured by public clouds that are very famous for on-demand, online, off-premise, consolidated, shared, automated, virtualized, and containerized IT services. In effect, IT organizations have to offer Shadow IT capabilities and benefits without the identified risks. Herein the celebrated role and responsibility of cloud brokerage solutions are vividly prescribed to provide Shadow IT capabilities yet without the articulated risks. With IBM cloudMatrix, IT organizations can devise a pragmatic approach to discover existing resources, provide visibility to new resources, and offer an equivalent alternative to Shadow IT. Organizations can start small and then extend capabilities and functionality as desired.

Establishing and Managing Multi-cloud Environments There are integration engines enabling distributed and different clouds to find, bind, and leverage one another's unique feats and features. Clouds are increasingly federated to accomplish special business needs. Clouds are being made interoperable through technology-centric standardization so that the vision of the Intercloud is to see the reality sooner than later. There are a few interesting new nomenclatures such as open, delta, and interoperable clouds. Apart from the virtualization dogma, the era of containerization paradigm to flourish with the industry-strength standards for containerization are being worked out. The Docker-enabled containerization is to have containerized applications that are very famous for portability for fulfilling the mantra of "make once and run everywhere." Developing, shipping, deploying, managing, and enhancing containerized workloads are made simple and faster with the open-source Docker platform. All these clearly indicate that disparate and distributed cloud environments are being integrated at different levels in order to set everything right for the ensuing days of people-centric and knowledge-filled services for achieving varying personal, social, and professional needs of the total human society. There are several business imperatives vehemently insisting for the onset of hybrid and multi-cloud environments. It is visualized that geographically distributed and different cloud environments (on-premise clouds, traditional IT environments, online, on-demand and off-premise clouds) need to be integrated with one another in order to fulfill varying business requirements.

Having watched and analyzed the market sentiments and business environments, it is safely predicted that multi-cloud environments will become the new normal in the days to unfurl. We need industry-strength and standardized integration and orchestration engines, multi-cloud management platforms, and a host of other associated empowerments in order to make multi-cloud environments a worthy addition for next-generation business behemoths. Clouds bring in IT agility, adaptivity, and affordability that in turn make business more easily and expediently attuned to be right and relevant for their constituents, customers, and consumers. Cloud brokerage solution is being touted as the most significant entity for presenting a synchronized, simplified, and smart front end for a growing array of heterogeneous generic as well as specific clouds. That is, cloud consumers need not interact with multiple and differently enabled clouds. Instead, users at any point of time from anywhere just interact with the cloud broker portal to get things done smoothly and in time with just clicks.

In conclusion, the role of a cloud broker is to significantly transform IT service delivery while ensuring the much-demanded IT agility and control. A cloud broker enables cloud consumers to access and use multiple cloud service providers (CSPs) and their distinct services. Further on, a cloud broker can also take care of the service delivery, fulfillment, API handling, configuration management, resource behavior differences, and other complex tasks. The broker facilitates users to take informed decisions for selecting cloud infrastructures, platforms, processes, and applications. This typically includes the cost, compliance, utility, governance, audibility, etc. Cloud brokers simplify the procedures and precipitate the cloud adoption and adaption. The IT operating model is bound to go through a number of transformations and disruptions through the smart leverage of cloud brokerage solution. The cloud complexity gets nullified through cloud brokerage solution. In short, the digital, API, idea, and insightful economy and era are bound to go through a radical transformation through cloud services and brokerage solutions.

A cloud service broker operationalizes best execution venue (BEV) strategies, which is based on the notion that every class of IT-related business need has an environment where it will best balance performance and cost and that the IT organization should be able to select that environment (or even have the application select it automatically). Brokers thus enable any organization to create the “right mix” of resources for its hybrid IT environment. The strategic goal of more with less is to get accentuated with all the accomplishments in the cloud space. Cloud users expect to be able to make decisions about how and where to run applications and from where to source services based upon workload profile, policies, and SLA requirements. As the worlds of outsourcing, hosting, managed services, and cloud converge, the options are growing exponentially. BEV strategies enable users to find the most suitable services to meet their needs. The cloud broker is the key element toward operationalizing this approach.

We have detailed how next-generation cloud broker solutions are to do justice to the abovementioned hybrid IT requirements. In the following sections, we are to detail how IBM cloudMatrix is emerging as the strategic software suite for the cloud brokerage needs.

11.9 Benefits of Having a Cloud Broker

- *Reduce the costs of cloud services* (30–40% estimated savings by using cloud brokers)
- *Integrate multiple IT environments* – existing and cloud environments, e.g., establish hybridity – as well as integrate services from multiple cloud providers
- *Understand what public cloud services are available* via a catalog
- *Policy-based service catalog* populated with only the cloud services that an enterprise wants their employees to purchase
- *Unified purchase cloud services* (broker) and help those services selected (by clients) better together
- *Assess current applications for cloud readiness*
- *Ensure cloud services meet enterprise policies*
- *Ensure data sovereignty* laws are followed
- Cloud brokers
  - *Cover all layers of the cloud stack* (IaaS, PaaS, and SaaS)
  - *Offer multiple deployment models*: on-premises (local) and off-premises (dedicated or shared). IBM supports all of these "as a service" deployment models but does not currently offer a traditionally licensed software product.

11.10 Conclusion

The cloud technology has matured and opened up newer possibilities and opportunities in the form of pragmatic hybrid clouds that in turn comprises elegant private clouds and elastic public clouds. Organizations are increasingly leaning toward hybrid clouds in order to reap the combined benefits of both public and private clouds. This document has explained the unique advantages to be accrued out of hybrid clouds. Enterprises considering the distinct requirements of their workloads are consciously embracing hybrid clouds. There are several hybrid cloud service providers in the market with different capabilities, and this document has faithfully articulated the various competencies of those hybrid cloud service providers in order to educate worldwide corporates to take an informed decision.

Appendix

VCE's Vblock systems provide a complete integrated solution for virtualization, storage, computing, and networking. Vblock is an engineered, manufactured, managed, and supported converged infrastructure that is ready to be deployed in your data center.

Vblock enterprise-class capabilities include management, performance, security, multitenancy, high availability, and backup. Vblock can easily scale out or up to meet all your business growth needs and protect your IT investment.

The Cisco ONE Enterprise Cloud Suite product portfolio consists of the following tools:

- Cisco Prime Service Catalog: Use a graphical approach to joining application elements with business policies and governance that can be consumed from a modern storefront portal.
- Cisco Intercloud Fabric for Business: Get consistent, highly secure hybrid cloud connectivity and workload mobility.
- Cisco UCS Director: Reduce data complexity and increase IT flexibility with unified infrastructure provisioning and management.
- Cisco Virtual Application Container Services: Rapid provisioning of virtual network services delivered with Cisco UCS Director infrastructure containers.

Microsoft

Microsoft is one of the few vendors to offer a true hybrid cloud solution. There are three core products: Azure, Windows Server, and Microsoft System Center. The company has proven itself as an on-premise provider, and its reputation is growing as a public cloud provider as well. Another big reason Microsoft takes the crown as the top hybrid cloud vendor is its flexibility and integration with existing product lines. The Windows Azure Pack covers most of the bases regarding IaaS, DBaaS, and PaaS. Microsoft shops will especially make use of the management capabilities of SQL Server as well.

Amazon

Amazon's Amazon Web Services (AWS) division is hands down, the juggernaut of the public cloud space. The massive number of customers on Amazon's platform and the range of tools and features available make it one of the top contenders in the cloud space.

AWS is known as a public cloud solution and does not provide all the required components for a full private cloud implementation. However, Amazon does offer integrated networking via the Amazon Virtual Private Cloud (Amazon VPC) and, via a group of partners, Direct Connect as part of its solution. Other Amazon partners provide backup and private storage, data integration, security, and configuration management. Combining AWS capabilities with those of partners like NetApp, F5, Splunk, Trend Micro, and Chef makes for a top-end hybrid cloud deployment.

VMware

VMware is still relatively new to the cloud space, but its depth of experience with virtualization and vendor-agnostic approach makes it a fierce competitor. VMware's approach to hybrid cloud is almost the opposite of AWS's, in that it's known for its private cloud products and utilizes a network of partners to deploy a fully hybrid solution. The private cloud portion is powered by VMware's vSphere. The "public" aspect of VMware's hybrid solution is vCloud Air – made available through the vCloud Air ecosystem of several thousand partners, with companies like CenturyLink and Claranet leading the charge.

Google

Google competes primarily with AWS and Microsoft Azure in the public cloud space, with its Google Cloud Platform. Like AWS, Google relies on a deep partner network to help fill out its hybrid cloud solution, but the size and customer base of Google Cloud Platform earned it a top spot on this list. With its background in data, Google tools like BigQuery are useful additions for the data-savvy ops team. And, given that Google shares many of the same partners that AWS utilizes in its hybrid cloud, users can expect similar types of integrations to be available.

Rackspace

Rackspace is another hybrid cloud vendor that works with a host of other vendors and products. Known for its focus on infrastructure, Rackspace offers dedicated database and application servers and dedicated firewalls for added security. Rackspace's hybrid cloud solution is held together by RackConnect, which essentially links an organization's public and private clouds. While it does offer VPN bursting and dedicated load balancing, Rackspace's catalog of additional tools and applications isn't as comprehensive as some of the competition.

Hewlett Packard Enterprise

HPE's Helion offering is focused on what it calls the Right Mix, where businesses can choose how much of their hybrid strategy will be public and how much will be private. HPE's private cloud solutions have a strong basis in open technologies, including major support for OpenStack. However, the company also leverages its partnerships with AWS and Microsoft Azure, among others, to provide some of the public cloud aspects of its hybrid cloud offering.

EMC

EMC's strength in hyper-convergence and plethora of storage options make it a good vendor for operation-heavy organizations who like to play a part in building out their own solutions. In terms of hyper-convergence, EMC has made many strides in the hardware space with its hardware solutions such as the VCE VxRack, VxBlock, and Vblock solutions. The company also offers a ton of security options but still relies on partners to provide the public cloud end of the deal.

IBM

IBM's Bluemix hybrid cloud is a valuable option, thanks to its open architecture, focus on developer and operations access, and catalog of tools available through the public cloud. Organizations looking to more effectively leverage data will find Watson and the IoT tools especially helpful. Using a product called Relay, IBM is able to make your private cloud and public cloud look similar, increasing transparency and helping with DevOps efforts. The company's admin console and syndicated catalog are also helpful in working between public and private clouds.

Verizon Enterprise

What many in IT don't realize is that most of the major telecom providers have cloud offerings of their own. Verizon Enterprise, the business division of Verizon, offers three customizable cloud models including a hybrid solution. Verizon Enterprise has a strong product in terms of disaster recovery and cloud backup. It also has a cloud marketplace and offers authorized Oracle integrations on Verizon cloud deployments.

Fujitsu

Fujitsu is another hybrid cloud provider built on another vendor's offering – in this case Microsoft Azure. Fujitsu Hybrid Cloud Services (FHCS) are a combination of Fujitsu's Public S5 cloud, running on Azure, and a private cloud, which is powered by Microsoft Hyper-V, and can be deployed on client side or in a Fujitsu data center. The offering provides standard tools like workload bursting, as well as the ability to split a workload by geography.

CenturyLink

CenturyLink is another telecom company that provides cloud services. The company advertises its service as a public cloud that is "hybrid-ready." Since it basically only provides the public cloud portion of a hybrid cloud deployment, CenturyLink is focused heavily on integrating with existing systems. Automation and containerization tools make it a good fit for shops that are exploring DevOps.

NTT

Japanese telecom giant NTT (Nippon Telegraph and Telephone) might fly under the radar by most IT leaders' standards, but it shouldn't be overlooked. The company's hybrid cloud solution is focused on security and privacy, with HIPAA, FISMA, and PCI compliance at the forefront. NTT's hybrid cloud has enhanced monitoring and additional security via Trend Micro. A plethora of optional features is available to further customize the deployment.

Cisco

Much like VMware, Cisco is known for its private cloud products and offers hybrid solutions through a partner network. Customers stitch their clouds together with the Cisco Intercloud Fabric, which allows users to manage workloads across their clouds. Cisco's partner network includes companies like Accenture, AT&T, and CDW, among many others.

CSC

Another up and comer in the hybrid cloud space is technology and professional services provider CSC. CSC's BizCloud is its private cloud component, and it partners with companies like AWS to provide the public cloud layer. CSC's big focus is on its Agility Platform, which connects different clouds together. The company uses adapters to make it easy to work with multiple providers.

Hitachi

Hitachi offers cloud storage on demand and compute as a service via its Hitachi Data Systems (HDS) division. Solutions are offered in outcome-based service-level agreements with a focus on customer choice. Hitachi also offers convergence tools and is a gold member of OpenStack, which signifies its commitment to open technologies.

Bibliography

1. http://www-03.ibm.com/software/products/en/cloudbrokerage About IBM cloudMatrix
2. The organisational impact of Bimodal IT, a white paper by Fujitsu, Japan, 2016
3. Orchestrating Hybrid IT, a white paper by Fujitsu, 2016
4. Hybrid IT takes center stage, a Harvard Business Review, 2016
5. The future of the data centre in the age of Hybrid IT, a white paper by Fujitsu, 2016
6. The Rise of Hybrid IT, a report by IDG Connect, 2016
7. Building the business case for Hybrid IT, a white paper by Fujitsu, 2016

Index

A
Accelerated Climate Modeling for Energy (ACME) workflow, 132
Adaptive hybrid model (AHM), 126
Adaptive window, 125
Analytical model (AM), 119, 121–125, 127–131, 133
Ant colony algorithm (ACO), 163–164, 177, 179, 182, 188
Application-specific performance prediction system (APPS), 120, 121, 123, 127
Artificial intelligence (AI) models, 32, 51, 122, 126
ASKALON, 55, 62–64, 123
Aspen, 129–131
Assembly and annotation workflow, 151–153
Auto regressive (AR), 125, 126
Automated workload consolidation, 171

B
Bayesian network, 127
Best effort workflow scheduling, 73, 77, 78
Bi-objective workflow scheduling, 74–75
BioCloud, 138, 144–153
Biomarker workflow, 178, 194–196

C
Cloud brokerage, 200, 209–216
Cloud computing, 3, 9, 17, 24, 26, 27, 32, 35, 55, 66, 68, 85–87, 101, 103, 104, 120, 141, 143–145, 149, 164, 165, 167, 168, 172, 175, 202
Cloud governance, 211
Cloud integration, 200
Cloudlet, 88, 91, 93, 94, 98–100
Cloud management, 33, 200, 205, 207–210
CloudMatrix, 210–212, 215, 216
Cloud orchestration, 200, 202
Cloud service providers (CSPs), 24, 26, 31, 37, 42, 47–50, 66, 87, 199, 205, 207, 216, 217
CloudSim, 74–77, 85–95, 101, 166–171, 173, 174
Clustering engine, 96, 97
Cognitive computing, 30, 31, 51
Collaborative method, 180
Component optimization, 181, 182
Condor, 107, 111
Consumerization, 2, 3, 41
Containerization, 3, 25, 28, 29, 34, 43, 51, 52, 201, 215, 220
Cost-based scheduling, 159–166
Customer-facilitated cost-based scheduling, 164
Cyber-physical systems (CPS), 7, 8

D
DAGMan, 107, 111
Data center (DC), 3, 5, 9, 16, 18, 25, 26, 29, 33, 35–38, 40–44, 46, 49–53, 55, 86, 92, 165, 167, 169–171, 200, 201, 204, 205, 207, 208, 212, 217, 220
Data center broker, 88
Deadline constrained, 74, 75, 159–160, 168
Digitalization intelligence, 2–22
Digitization, 2–6, 8, 12–18, 22, 44
Digitized elements, 8
DIMEMAS, 123

G. Kousalya et al., *Automated Workflow Scheduling in Self-Adaptive Clouds*,
Computer Communications and Networks, DOI 10.1007/978-3-319-56982-6

Directed Acyclic Graph (DAG), 24, 55, 64–70, 90, 95, 101, 103, 106, 107, 109, 113, 140, 148–150, 157, 169
Distributed computing optimization, 45, 186–188
Dynamic scheduling, 24, 66, 68, 70, 75

E
Energy-based scheduling, 165–171
ExomeSeq workflow, 151, 152

F
Fast Agent's System Timer (FAST), 124
Fog computing, 31
Free load profile (FLP), 123
FREERIDE-G, 124

G
GAMMA model, 123
Genetic algorithm (GA), 70, 127, 177, 179, 188–190
Graphical-based systems, 56

H
Hybrid cloud, 17, 75–77, 162, 163, 199–211, 213, 217–220
Hybrid Cloud Optimized Cost (HCOC) schedule, 73, 162–163
Hybrid IT, 199–221
Hybrid performance prediction systems (HPPS), 127–128

I
Independent data tuples (IDTs), 122
Instance-based learning (IBL), 122, 126
Integrative method, 180

J
Job scheduler, 61, 62, 106, 126

K
Kepler, 55–58, 64, 183
Kickstart, 130, 131

L
LaPIe, 123
Local execution engine, 61, 62, 106

M
Machine learning models (MLMs), 119, 121, 122, 125–128, 131, 133, 194
Mapper, 61, 95, 106, 109
Modified workflow life cycle, 180–183
Monitor-Analyze-Plan-Execute (MAPE-K), 131
Moving average (MA), 125
Multi-objective, 24, 66, 75, 76, 149, 168, 178, 179
Multi-objective workflow scheduling, 75–79
Multiple MLM multiple AM (MMMA), 128
Multiple MLM single AM (MMSA), 128
Multiple workflows, 68, 76, 77, 142, 163

N
Network function virtualization (NFV), 36, 37, 41, 42, 49, 52
Network Weather Service (NWS), 124–126

O
OpenStack, 27, 33, 104–109, 117, 146, 210, 221
Optimization levels, 177, 180–183, 188
Optimization phase, 177, 180, 181, 196
Optimization plug-in, 177, 183–185, 190–196

P
PANORAMA, 120, 128–133
Parallel workflow execution, 187
Parameter optimization, 179, 181, 183, 188–196
Particle swam optimization (PSO), 73, 160–162, 166, 179, 188
Pegasus, 55, 61–62, 64, 104–117, 129, 130, 142, 182
Pegasus workflow management system, 61, 104–107, 112, 114, 115, 117, 129, 130
Predicting query runtime (PQR), 126
Predictive analytics, 10–12
Prescriptive analytics, 10–12
Process dispatch, 184, 185
Proteomics workflow, 178, 192–193

R
Radial basis function neural network (RBF-NN), 127
Remote execution engine, 61, 62, 106
Rensselaer's Optimistic Simulation System (ROSS), 129, 130
Runtime optimization, 182

S
Scientific workflows, 24, 55–64, 66, 73, 76, 77, 79, 80, 85, 95, 103–105, 120, 121, 128, 130, 138, 140, 142, 143, 159, 165–168, 177, 178, 180, 182, 183, 188, 189
Script-like systems, 56
Simulation, 59, 73–77, 79, 85, 95, 97, 101, 129, 130, 132, 133, 166, 169, 172, 188
Single MLM single AM (SMSA), 128
Single-objective, 24, 66, 74, 179, 180
Single workflow, 68, 183
Sliding window, 125
SNS workflow, 132
Software-defined cloud environments (SDCEs), 23–53
Software-defined compute (SDC), 34, 45, 52, 53
Software-defined networking (SDN), 5, 37–44, 49, 52
Software-defined storage (SDS), 25, 44–49
Spatial-temporal correlation models (STCs), 122
STAMPEDE framework, 131
Static scheduling, 24, 66, 68–70, 75

T
Task profiling model (TPM), 123
Task selection priority and deadline (TPD), 164–166
Taverna, 55, 58–60, 64, 177, 183–187, 190, 191, 196
The Internet of Things (IoT), 6–9, 31, 51, 219
Topology optimization, 182
Triana, 55, 60–61, 64, 141

V
Virtual machine (VM), 9, 24, 25, 27, 31–37, 42, 43, 45, 52, 66, 79, 85–89, 91–94, 96, 116, 142, 145, 146, 149, 157, 163–165, 167–171, 206, 209–211, 213
Virtualization, 3, 9, 11, 12, 14, 21, 24, 25, 27, 29–31, 34–38, 41, 42, 45, 46, 50–53, 66, 72, 86, 104, 201, 209–211, 215, 218

W
Workflow challenges, 79, 80, 137–153
Workflow engine, 60, 71, 80, 96–98, 111, 112, 117, 137, 139, 141, 142, 158, 181, 186
Workflow ensembles, 68, 74, 75
Workflow life cycle, 137–142, 153, 177, 180–183
Workflow Management Coalition (WMC), 71, 158
Workflow management systems (WMS), 55–64, 80, 85, 89, 90, 95, 97, 104–107, 112, 114, 115, 117, 129, 130, 143, 144, 183
Workflow mapper, 95
Workflow mapping, 109, 137, 138, 141
Workflow metadata and provenance, 137–139, 142
Workflow model, 56, 67–69, 85–101
Workflow optimization, 95, 175, 177–181, 183, 184, 188, 192
Workflow prediction, 119–133
Workflow reference model, 71, 157, 158
WorkflowSim, 85, 95–101
Workflow scheduler, 79, 95, 96, 98
Workflow scheduling (WS), 24, 31, 33, 52, 65–80, 90, 95, 103–117, 128, 144, 157–159, 162, 163, 165, 167, 170, 175
Workflow structure, 67, 89, 90, 112, 119, 120, 159, 182

X
XBaya, 140