Enterprise Content and Search Management for Building Digital Platforms

Enterprise Content and Search Management for Building Digital Platforms

Shailesh Kumar Shivakumar

IEEE computer society

IEEE PRESS

WILEY

Published by John Wiley & Sons, Inc., Hoboken, New Jersey.
Published simultaneously in Canada.

Limit of Liability/Disclaimer of Warranty: While the publisher and author have used their best efforts in preparing this book, they make no representations or warranties with respect to the accuracy or completeness of the contents of this book and specifically disclaim any implied warranties of merchantability or fitness for a particular purpose. No warranty may be created or extended by sales representatives or written sales materials. The advice and strategies contained herein may not be suitable for your situation. You should consult with a professional where appropriate. Neither the publisher nor author shall be liable for any loss of profit or any other commercial damages, including but not limited to special, incidental, consequential, or other damages.

For general information on our other products and services or for technical support, please contact our Customer Care Department within the United States at (800) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley also publishes its books in a variety of electronic formats. Some content that appears in print may not be available in electronic formats. For more information about Wiley products, visit our web site at www.wiley.com.

Library of Congress Cataloging-in-Publication Data:

Names: Shivakumar, Shailesh Kumar, author.
Title: Enterprise content and search management for building digital platforms /
Shailesh Shivakumar.
Description: Hoboken : Wiley, 2016. | Includes index.
Identifiers: LCCN 2016032953 (print) | LCCN 2016048578 (ebook) |
ISBN 9781119206811 (paperback) | ISBN 9781119206828 (pdf) |
ISBN 9781119206835 (epub)
Subjects: LCSH: Management–Technological innovations. | Digital media–Management. |
Multimedia systems–Management. | Performance technology. |
BISAC: COMPUTERS / Web / Site Design.
Classification: LCC HD30.2 .S558 2016 (print) | LCC HD30.2 (ebook) |
DDC 658.4/038011–dc23
LC record available at https://lccn.loc.gov/2016032953

Printed in the United States of America.

10 9 8 7 6 5 4 3 2 1

To my parents, Shivakumara Setty V and Anasuya T M,
from whom I borrowed love and strength
To my wife, Chaitra Prabhudeva, and my son, Shishir,
from whom I borrowed time and support
To my in-laws, Prabhudeva T M and Krishnaveni B,
from whom I borrowed help and courage
and
To all my schoolteachers who bestowed lots of love and
knowledge upon me

Contents

Part 2 Advanced Content Management

Part 3 Enterprise Search Technologies

Preface

Disruption in digital technologies has opened up an entirely new realm of possibilities for enterprises. Harvesting new-age digital technologies can redefine the ways business is done online and can potentially give numerous possibilities to reengage with stakeholders such as consumers, partners, resellers, and others. Digital technologies enable enterprises to provide on-demand, customer-centric, personalized, contextual, and meaningful content from anywhere, anytime, on any device. Digital-enabled business models reshape customer experiences and form the key differentiators. As a result, the digital user will be meaningfully engaged bringing in productivity, loyalty, and long-term relationship. On the B2B front, digital technologies have also opened up new realms of possibilities through process optimizations, enterprise integrations, and other developments, and a digital technology ecosystem has reshaped infrastructure and operations sides of things through hardware consolidation and cloud enablement.

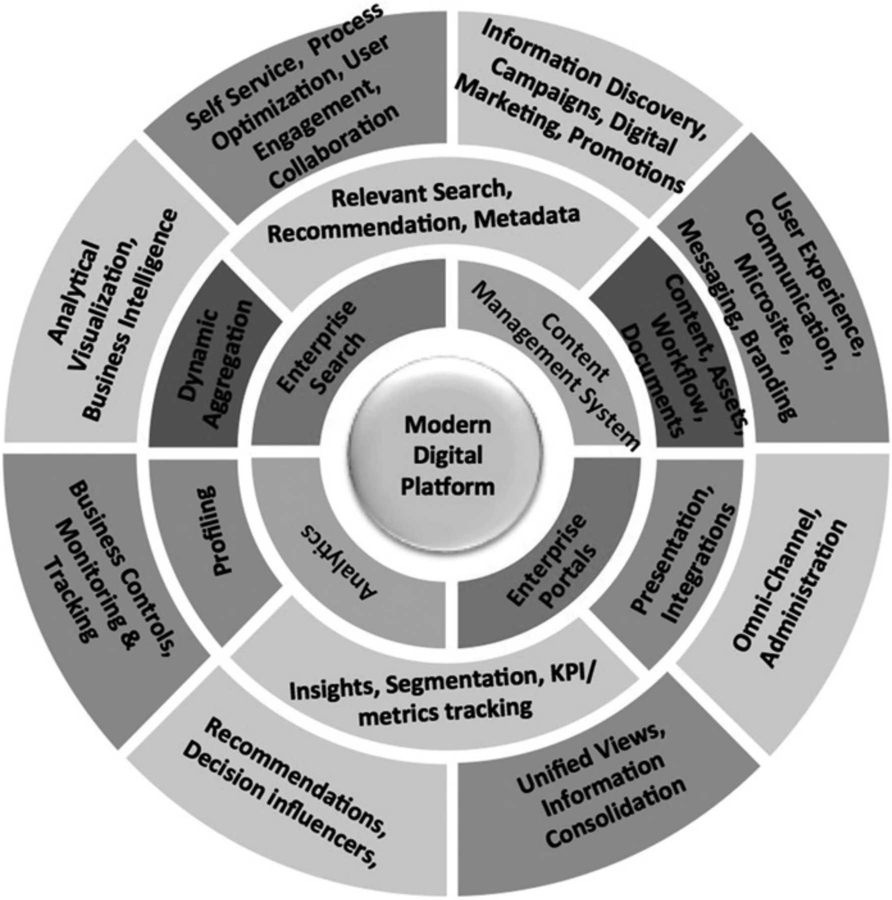
Digital technologies are disrupting most of the business domains, technology ecosystems, and business processes. Due to its wide range of benefits and long-term strategic impact and competitive advantages, enterprises across domains are embracing digital revolution. In today's hyper-connected world, word-of-mouth promotion is given preference over sponsored ads, Facebook "likes" count more than expert rating, and enterprises strive hard to convert a visitor into a brand advocate using digital technologies.

DRIVERS AND MOTIVATIONS FOR THE BOOK

Modern enterprises face multiple challenges in building a robust enterprise digital ecosystem. The challenges are multifold in nature and consists, among other things, of internal challenges concerning employee productivity, process optimization, information management, content management, and big data management. Coupled with these are external challenges such as, among others, Omni-channel customer engagement, social and collaboration integrations, personalized presentation, and competitive pressures.

Based on our experience, the most effective way to address these challenges is to provide a *robust information management system* consisting *of seamless relevant information discovery.* This book tries to address these two fronts by exploring various concepts in digital content management (for information management) and enterprise search (for efficient information discovery). This book takes a differentiated view through a combined focus on content management and enterprise search. During the process it aims to help organization build robust digital platforms using proven best practices, practical models, and time-tested techniques discussed in the book.

We can map the technology topics (CMS and Enterprise Search) discussed in this book with the modern digital platform's capability as shown in the following diagram:


Modern Digital Platform
Enterprise Search
Content Management System
Enterprise Portals
Analytics
Relevant Search, Recommendation, Metadata
Content, Assets, Workflow, Documents
Presentation, Integrations
Insights, Segmentation, KPI/ metrics tracking
Profiling
Dynamic Aggregation
Self Service, Process Optimization, User Engagement, Collaboration
Information Discovery, Campaigns, Digital Marketing, Promotions
User Experience, Communication, Microsite, Messaging, Branding
Omni-Channel, Administration
Unified Views, Information Consolidation
Recommendations, Decision influencers,
Business Controls, Monitoring & Tracking
Analytical Visualization, Business Intelligence

The first layer depicts the core digital technologies, namely ontent management system (CMS), enterprise search, portals, and analytics. The second layer maps the technical capabilities offered by corresponding technologies. CMS provides robust content management, workflows, documents, and asset management whereas search provides relevant search and recommendations. Both CMS and search would enable metadata-tagging capabilities. The outermost layer depicts the business capabilities enabled by corresponding technology capabilities. Content management enables intuitive user experience, communication, messaging, branding, and micro-site. Search and CMS combined enables promotions management, campaigns, and marketing and relevant information discovery capabilities.

The diagram depicts the importance of role played by CMS and enterprise search in a digital scenario. CMS and search form the information management backbone for a digital enterprise. The book tries to cover the capabilities discussed under CMS and search umbrella and relevant analytics capabilities wherever applicable.

Key Differentiators of the Book

The key differentiators and novel aspects of this book are summarized in the following list:

- **Wide coverage of modern methodologies and techniques:** We have covered emerging technologies such as micro services architecture in content management, CMS-based customer experience platform (CXP), Big Data search, semantic search, Omni-channel content enablement, JCR and CMIS standards, content analytics, SEO, and KPIs. We have detailed trends in CMS and enterprise search we have noticed and have provided good coverage of emerging trends. CMS is explored from security, infrastructure, and performance viewpoint as well.
- **Content frameworks:** The book covers many practically proven models and techniques related to CMS evaluation framework, content migration framework, search evaluation framework, and other aspects that can be used in real-world digital engagements. Comprehensive CMS, search, and DAM evaluation templates are given in Appendixes C, D, and F, respectively.
- **Elaborate content strategy discussion:** As content strategy forms the core of content management, we engage in an in-depth discussion of it in Chapter 2 along with a detailed case study. All chapters in Parts I and II are organized to realize various elements of content strategy discussed in Chapter 2. We have also provided a content strategy template in Appendix A to complement the concepts discussed in Chapter 2.
- **Case-study-based approach:** All core topics (such as templates, workflows, content security, performance, metadata, document management, content migration, and such) have detailed in-context case studies to provide the practical flavor to the topic discussion. The Online Wiley book support material section provides content case studies to explain the best practices used in real-world engagement. Case studies are used as tools to reinforce the theory concepts and provide practical applicability for them. Online support material

also has an elaborate end-to-end digital program case study covering CMS and enterprise search for a digital e-commerce platform.

- **Sample code and configuration:** We have provided sample code while discussing JCR migration concepts to elaborate the concept in Chapter 9. We have also given the configurations that can be used to address security vulnerabilities and optimize content performance in Chapters 11 and 12, respectively.
- **Reference architectures:** The book provides reference architecture for various CMS and search-based applications. Reference architecture of CMS-based customer experience platform, knowledge management system, digital marketing platform, and e-commerce platform are elaborated.
- **Proven best practices and checklists:** We have provided elaborate practically proven best practices while discussing key topics (such as content services, content security, templates, etc.). We also provided content management checklist in Appendix B section. Architects and managers for content and search engagements could use this.
- **Content integrations:** We have dedicated Chapter 7 to integrations with CMS providing details about optimal integration techniques with CMS.
- **Synergies between enterprise content and search:** This book tries to explore the synergies between enterprise content and search systems to build a robust digital platform. Metadata, taxonomy, SEO, analytics, and digital program management are explored from this dimension.
- **Practically proven models and best practices:** We discuss various models and best practices related to content such as template design, workflow design, and Omni-channel content design that are successfully employed in various practical engagements.
- **Architecture concepts:** There is an in-depth coverage of various architecture concepts for content management and digital search. Practitioners can use this as reference architecture in digital programs.
- **Reusable templates:** We have provided CMS evaluation template, search product evaluation, and content strategy template in the appendix sections. Readers can use it for content programs.

MAIN THEMES AND FOCUS AREAS

Main themes and focus areas of this book are:

- **Digital content management and enterprise search:** The primary focus areas of this book are digital content management (primarily Web content and digital assets through Web content management [WCM] concepts) and enterprise search. Wherever necessary, the book also elaborates other supporting systems/components such as digital asset management (DAM) systems, document management system, workflow management, and Web analytics, among others.

- **Technology and product agnostic view:** The concepts, methodologies, techniques, and best practices discussed in the book are product and technology agnostic. Wherever necessary, concrete examples are drawn from specific technologies and products to explain the concept.
- **Open source frameworks:** Many of the concrete examples are drawn from open source products. Some reference architectures are also developed using open-source components. The intention is to help readers leverage open-source technologies while creating digital systems.
- **Proven practical methodologies and best practices:** We have elaborated many proven models and best practices in areas such as content migration, CMS evaluation framework, content performance, content security, and such. This would help the content and search practitioners apply these frameworks and techniques.
- **Challenges and best practices:** While discussing core portal technologies such as integrations, content management, search, and others, we have discussed the commonly encountered challenges/pitfalls and the best practices.

CHAPTER ORGANIZATION AND TARGET AUDIENCE

The book is organized in three parts with 14 chapters. The online Wiley book support material section provides various supporting material such as content case study and end-to-end digital case study. Part I consists of six chapters that introduce reader to core concepts of content management. We look at content strategy, CMS basics, CMS architecture, templates and workflow, information architecture, taxonomy, and content metadata. Part II includes six chapters and extends the content management concepts and elaborates on topics related to integration, content standards, DAM and document management, content migration, CMS evaluation, content validation, content analytics, content security, content performance. Part III consists of two chapters and is mainly dedicated to discussing basics of enterprise search and advanced search.

We have provided six appendix sections: Appendix A provides a content strategy template, Appendix B provides a checklist for various content management activities, Appendix C is a CMS product evaluation template, Appendix D is the enterprise search product evaluation template, Appendix E provides sample Java code for adding a JCR node, and Appendix F provides an evaluation template for DAM platforms.

The following is the high-level summary of various chapters along with intended target audience:

Chapter	**Main topics**	**Target Audience**
Part I: Content Management Basics For Digital Platforms		
Chapter 1: Introduction to Digital Platforms	Enterprise digital ecosystem, enterprise content management concepts, digital strategy, content strategy, digital content management overview, enterprise search overview	Digital architects, enterprise architects, program managers, business analysts, and senior business executive
Chapter 2: Content Strategy	Content strategy overview, strategy challenges, strategy essentials, content characteristics, requirements elaboration, content strategy definition process, content strategy phases, content strategy elements, content strategy case study	Content architects, content strategists, CMS developers, and enterprise architects
Chapter 3: Basics of Content Management System	CMS drivers, CMS design principles, CMS attributes, CMS capabilities, Discussion of CMS systems (WordPress, Drupal, Joomla), and Apache Jackrabbit	Enterprise architects, content architects, and CMS developers
Chapter 4: Content Management System Architecture	CMS design and architecture, CMS architecture patterns, CMS value articulation framework, CMS solution component design, Multi-site design, content folder design, content URL design, localization design, CMS infrastructure design, content strategy realization in CMS, CMS reference architecture, customer experience platform design, knowledge management system design, and content marketing platform design	CMS architects, CMS developers and enterprise architects
Chapter 5: Templates and Workflows	CMS template design, authoring and presentation templates design and usage, template-user interface, template development case study, workflow design, workflow optimization, workflow case study	Content architects, content authors, and CMS developers
Chapter 6: Content Information Architecture, Taxonomy, and Metadata	Designing intuitive information architecture (IA), elements and goals of IA, taxonomy concepts, taxonomy and metadata, metadata types, metadata standards (Dublin Core and SKOS), metadata case study	Enterprise architects, content architects, content strategists, and information architects

Chapter	Main topics	Target Audience
Part II: Advanced Content Management		
Chapter 7: Content Integration and Content Standards	CMS integrations with security systems, TMS, search engine, portals, presentation engines, metadata systems, feeds, DAM, CMIS integrations. Content standards: HTML/XML, DITA, JSON, SCORM, Web service formats (REST and SOAP)	Content architects, Integration architects, CMS administrators, CMS developers, and enterprise architects
Chapter 8: Digital Asset Management and Document Management	DAM objects, architecting DAM system, DAM challenges, document management system capabilities, document management evolution and road map, document management case study	Content architects, CMS developers, content authors
Chapter 9: Content Migration	Migration drivers, migration design considerations, migration challenges, migration checklist, migration governance, migration automation, migration case study	Enterprise architects, CMS architects, CMS developers and program managers
Chapter 10: Content Maintenance – Validation, Analytics, KPI, SEO, and Evaluation	Content validation types, validation checklist, content analytics, content KPIs, content analytics design, content analytics case study, content SEO strategy, content SEO best practice, CMS evaluation framework	CMS architects, CMS developers, CMS QA team and program managers
Chapter 11: Content Security	Content security vulnerabilities, XSS, CSRF, denial of service, clickjacking, generic content security scenarios, SSO, penetration testing, security best practices, security testing case study	Content architects, security architects, enterprise architects
Chapter 12: Content Infrastructure and Performance Optimization	Content performance optimization, CMS-level performance optimization, infrastructure-level performance optimization, content performance KPIs, content performance testing	Content architects, enterprise architects, performance engineers, CMS developers

Chapter	Main topics	Target Audience
Part III: Enterprise Search Technologies		
Chapter 13: Introduction to Enterprise Search	Enterprise search drivers, search overview, search trends, search evolution, search capabilities, basic search features, advanced search features, Apache Solr and ElasticSearch features	Enterprise search architects, information architects, enterprise architects, and search developers
Chapter 14: Advanced Enterprise Search	Federated search, features, architecture and challenges of federated search, relevancy rank adjustment, personalized search, semantic search, semantic search process, people search, Big Data search	Search architects, enterprise architects, program managers, and search developers

DECLARATION

- Utmost care has been taken to ensure the accuracy and uniqueness of the book content. Any inaccuracies or inconsistencies are entirely my own. If you think any corrections are needed, or for any other feedback, please write to Shailesh.shivakumar@gmail.com
- In a few chapters I have used the features of popular and open-source WCM products to explain the concepts. The explanation is for educational purposes only and should not be considered as a product or technology recommendation or evaluation. The CMS plugins and modules used to illustrate examples and concepts are also for educational purposes only; they should not be interpreted as recommendations or evaluations. Comprehensive evaluation template and framework are provided in the appendix section.
- All open-source tools mentioned are in public domain as open source at the time of writing of this book.
- I acknowledge the trademarks of all products, technologies, and frameworks being used in this book.
    - WordPress, Joomla, Drupal, and dotCMS are registered trademarks and are the legal property of their respective owners.
    - Documentum is a registered trademark of EMC Corporation.
    - Oracle, Oracle Access Manager, WebCenter, WebLogic, OHS, and Java are registered trademarks of Oracle and/or its affiliates.
    - Synaptica is the registered trademark of Synaptica, LLC.
    - SiteMinder is the registered trademark of CA Technologies.
    - WebSphere, Tivoli Access Manager, IBM WCM, IHS, and DB2 are registered trademarks of IBM and/or its affiliates.
    - AEM and CQ5 are registered trademarks of Adobe and/or its affiliates.

- FAST, SharePoint, and SQLServer are registered trademarks of Microsoft and/or its affiliates.
- Liferay is a registered trademark of Liferay and/or its affiliates.
- Intel is a registered trademark of Intel Corporation.
- All other trademarks or registered trademarks are the legal property of their respective owners.

Acknowledgments

I am blessed to be surrounded by knowledgeable colleagues, subject matter experts, and friends who are happy and willing to help. I would like to acknowledge them and show my gratitude for their immense help, incredible support, and cooperation.

I would like to convey my sincere and heartfelt thanks to Elangovan Ramalingam, Arun Sugumar, Ashwin Raju, Verma V.S.S.R.K, Subramanian Narayanan, Ajay Kumar Sharma, Shikha Mahajan, Aanchal Sikka, and Nagarajan Kuppuswamy for their valuable inputs and review comments. I feel blessed in the company of these gifted colleagues.

I would also like to convey my sincere thanks to managers Shankar Bhat, Rahul Krishan who encouraged and supported me in all my initiatives.

My sincere and heartfelt thanks to Professor Dr. Viraj Kumar for his moral support and patience. He has been a continuous inspiration to me.

I really appreciate the efforts and support given by Rahul Krishan and Sreenivasa Kashyap at Infosys for ensuring timely reviews and approvals of this book.

I would also like to recognize and thank Dr. P. V. Suresh for his constant encouragement and immense support.

My special thanks to Mary Hatcher, Melissa Yanuzzi, Brady Chin, Allison McGinniss, Alex Castro, and the editors, designers, and publishing team at the IEEE and Wiley for providing all necessary and timely support in terms of review, guidance, and regular follow-ups. The team at Wiley took special care in design to make this book look beautiful.

About the Author

Shailesh Kumar Shivakumar is a Senior Technology Architect at Infosys Technologies Limited with over 15 years of industry experience. His areas of expertise include digital technologies, software engineering, Java enterprise technologies, performance engineering, and digital program management. He is a Guinness world record holder of participation for successfully developing a mobile application in coding marathon. He has four patent applications including two US patent applications in the area of Web and social technologies.

He was involved in multiple large-scale and complex digital transformation programs for Fortune 500 clients of his organization. He also provided on-demand consultancy in performance engineering for critical projects across various units in the organization. His has hands-on experience on breadth of technologies including Web technologies, digital technologies, and database technologies and has worked on multiple domain areas such as retail, manufacturing, e-commerce, and avionics, among others. He was the chief architect of an online platform that won "Best Web Support Site" award among global competitors.

He is the author of two technical books: *Architecting High Performing, Scalable and Available Enterprise Web Applications* and *A Complete Guide to Portals and User Experience Platforms*. He is a regular blogger at Infosys Thought Floor, and many of his technical white papers are published in Infosys external site. He has delivered two talks at Oracle JavaOne 2013 conference on performance optimization and project management and has presented a paper at IEEE conference. He also headed a center-of-excellence for digital practice. He led multiple thought-leadership and productivity improvement initiatives and was part of special interest groups (SIG) related to emerging Web technologies at his organization.

He holds numerous professional certifications including TOGAF 9 certification, Oracle Certified Master (OCM) Java Enterprise Edition 5, Sun Certified Java Programmer, Sun Certified Business Component Developer, IBM Certified Solution Architect – Cloud Computing, IBM Certified Solution Developer – IBM WebSphere Portal 6.1, and many others.

He has won numerous awards including prestigious Infosys Awards for Excellence 2013–14 “Multi-talented thought leader" under “Innovation – Thought leadership” category, “Brand ambassador award” for MFG unit, “Best employee award,” delivery excellency award, and multiple spot awards and received honor from executive vice chairman of his organization. He is featured as “Infy star” in Infosys Hall of fame and recently led a delivery team that won the “best project team” award at his organization.

He holds an engineering degree in computer science and has done executive management program from Indian Institute of Management, Calcutta. He lives in Bangalore, India.

About the Companion Website

This book is accompanied by a companion website:

www.wiley.com/go/shivakumar/enterprisecontent

The website includes:

- Additional Book Chapters
  - Chapter 15 – Content Management Case Studies
  - Chapter 16 – Digital Case Study: Building a Modern Digital E-commerce Platform Using CMS and Search

- Appendices A-F
  - A – Content Strategy Template
  - B – Content Management Checklist
  - C – CMS Evaluation Template
  - D – Enterprise Search Evaluation Template
  - E – Java Code for Adding an Image Node to Jackrabbit Workspace
  - F – DAM Evaluation Template

Part 1

Content Management Basics for Digital Platforms

Chapter 1

Introduction to Digital Platforms

Digital transformation is a part of a technology road map for most of the enterprises (especially the ones operating in the B2C domain). Digital technologies are disrupting the way end-users engage with online channels and redefining the way enterprises do business online. Enterprises strive to provide differentiated experiences to its digital consumers. Digital channels personalize information at all levels to convey the right message that resonates with individual end-user's personality. Disruptive digital technologies build a participative digital platform enhancing user's digital journey and consistently aim to convert a visitor into a brand advocate. As digital experience assumes the prime focus of online platforms, it becomes imperative to understand various aspects of digital technologies and use them to their fullest potential.

This chapter provides insights into digital ecosystem and lays the foundation for core digital technologies including content management and enterprise search that are the main focus areas of this book. The chapter also briefly introduces the concepts related to digital content management and enterprise search and their value addition to the digital platforms. We will also look at the elements of enterprise digital strategy and digital content strategy. All chapters in Parts I and II are developed to provide in-depth details of various components needed to realize the content strategy. This chapter provides a foundation for upcoming chapters.

Chapter organization: We start by looking at various elements of an enterprise digital ecosystem, such as opportunities, challenges, and digital capabilities. We will learn how digital technologies are making a fundamental impact on the traditional business functions. We then look at enterprise content management (ECM) to understand the big picture of overall content management. Specifically we will learn about ECM features, applications, and differences between ECM and WCM. The next section discusses various elements of enterprise digital strategy and content strategy. The strategy discussion lays the foundation for concepts in remaining chapters for this book and helps us understand the role of each content component (and corresponding

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

book chapter) in the content ecosystem. We conclude with a high-level discussion of content management and enterprise search along with its value-added.

Digital architects, enterprise architects, program managers, senior business executives, and business analysts will find this chapter useful.

1.1 ENTERPRISE DIGITAL ECOSYSTEM

A robust digital strategy involves leveraging digital technologies to actively collaborate, engage, and maintain long-term relationship with all key stakeholders and continuously optimize their experience with the digital channels. In this section we will look at the core elements of digital ecosystem from technology and functionality standpoint and understand the big picture of enterprise digital ecosystem components.

The digital transformation provides a wide variety of opportunities across a wide spectrum of business domains. We will look at various opportunities opened up by digital technologies, including a quick look at the key challenges while implementing them. Digital Experience, Digital marketing, digital commerce, and SMAC (Social, Mobile, Analytics, and Cloud) form a major chunk of capabilities required for customer-facing B2C enterprises. Collaboration, engagement, and innovation forms a vital portion of partner relationship through B2B channels. For internal stakeholders such as employees, productivity improvements, collaboration, operational efficiency, and active employee engagement form crucial aspects of B2E (Business-to-employee) channels. In the section that follows we discuss the topics related to digitally driven capabilities and technology enablers.

Digital Opportunities for Enterprises

The success of modern online platforms experiences is based on the customer experience with digital platforms. Organizations are transforming online channels to provide superior customer experiences to retain and grow an existing customer base. Key organization focus areas such as digital transformation, digital commerce, digital marketing, social and collaboration enablement, and legacy modernization comprise the main elements of a digital ecosystem.

Going digital provides the following opportunities to enterprises:

- Provide consistent Omni-channel experience to provide content anytime, anywhere, on any device.
- Enhance information discovery process and provide personalized and relevant information for a given context.
- Leverage social exchange and collaboration to engage, collaborate, and influence customer behavior. Monitoring and listening to social conversations helps in better branding, improved product quality management, and for achieving effective customer service.

- Gather real-time insights into customer behavior through analytics. Customer behavior, navigations, actions, purchase patterns can be used for enhancing and personalizing user experience.
- Provide contextual, personalized content and recommendations to drive customer's decision making.
- Leverage data from all customer interactions to provide highly engaging customer support and user experience. Move from transactional model to long lasting relationship model with customers.
- Provide unified and consolidated view of customer to design better marketing campaigns and sale offers, loyalty offers, and personalized promotions.
- Enable a self-service model across various customer interaction points and optimized operations.
- Convert the site traffic into sales through personalized and engaging experiences.
- Enhance productivity of the workforce through self-service tools and simplified processes and productivity-enhancing tools and accelerators.
- Streamline platform support, maintenance, and operations through intuitive dashboard-based real-time monitoring. Adopt agile delivery model to shorten the time to market.
- Provide "on the go" and on-demand services and optimize the infrastructure cost through cloud operations.
- Optimize administration features for managing multiple sites, multiple languages, and monitoring and multilingual content.
- Effectively comply with regulation and compliance policies through optimal usage of content systems and process optimization.
- Consolidate various views into a unified platform and eliminate heterogeneous content silos that lack integrated business view.
- Provide simplified, optimized, and automated processes for better user experience.
- Provide an collaboration platform enabling users to share the knowledge and artifacts to effectively use the collective knowledge.

As we can see, digital technologies have huge potential to disrupt enterprises and enhance user experiences. The next section presents prominent challenges digital enterprises face. Digital capabilities to address these challenges are presented in subsequent sections.

Challenges in Modern Digital Enterprises

The following list presents some of the key challenges that digital enterprises are facing today. These business challenges stand in the way of building a robust digital platform.

a. **Enterprise integrations**

Due to explosive growth of enterprise data coupled with increased diversity of enterprise systems, integrating structured and unstructured data is becoming a daunting task. Making sense of such data would also be challenging in the absence of optimal integrations. Enterprises should collate data from various enterprise sources to allow digital platform to provide meaningful data to various users. Extracting meaningful data from various structured and unstructured sources is one of the key challenges for digital enterprises.

b. **Enterprise process modeling and optimization**

As enterprises enter new geographies, the complexity of business rules and underlying business processes tend to increase. Modeling these business processes through workflows represents another key challenge.

c. **Matching digital consumer and market expectations**

Tech-savvy consumers and increased competition pose a new set of challenges to the digital enterprises. Modern digital consumers expect dynamic, Omni-channel, and rich user experiences that are highly responsive and interactive. They want to be active participants in the collaboration, knowledge creation, and other similar processes. Digital customers share their experiences in online forums that can influence the community. Managing customer and market expectations and streamlining underlying processes/operations is another major challenge.

d. **Collaboration challenge**

Organizations face challenges in creating collaborative and self-service platforms due to the lack of standard integration interfaces. Bringing the cultural shift and associated processes to drive collaboration is one of the key challenges.

e. **Consolidation challenge**

Consolidating functionality, technology stack and content spread across various systems, geographies, and formats is another key concern. Eliminating content redundancies and increasing content reusability are some of the key drivers for digital transformation.

Additionally enterprises would face other functional challenges such as nonstandard interfaces, varied compliance requirements, and technology challenges related, among other things, to content duplication, content migration, and content distribution.

Now that we have looked at various opportunities associated with digital technologies and common enterprise challenges, we turn our focus to the means of filling the gap. In the next section we examine the role of digital technologies in this regard.

Enterprise Digital Capabilities

Here is a look at various digitally enabled capabilities that can effectively address the challenges discussed earlier.

The main enterprise digital technology capabilities are listed in the Figure 1.1. The key digital technologies in Figure 1.1 are explained below:

- Access control and security forms core security requirement for all digital platforms. This includes authentication of various forms, fine-grained authorization, user registration, Web SSO, federated SSO, account reset, user administration, user provisioning, certification management, directory services, and similar security components.
- Presentation and personalization is the user experience layer of the digital platform. It mainly includes presentation components such as responsive pages and widgets as indicated in the Figure 1.1. This layer mainly handles user experience management through immersive, compelling, and inspirational content through easy-to-use, friendlier-navigation, easy-to-discover, and easy-to-complete processes across the entire customer journeys. Delivery of personalized and localized content on all channels is one of the main capabilities in this category. Presentation portal modules render the pages to end-users based on personalization rules. Templates and layout modules are used for page construction. SEO and analytics modules track and monitor customers' online activities and provide real-time intelligence through intuitive and actionable reports to concerned stakeholders. This can be used in a variety of ways, such as optimizing user experience, customer support, fine-tuning customer campaigns, business KPI tracking, and so forth. Localization modules render the pages and content in user-specific locales. Responsive Web Design (RWD) is the main technology enabler for Omni-channel delivery.
- As business processes are increasingly being automated and optimized, business process management (BPM) is one of the main components to optimize business processes through process orchestration, data transformation, automation, and enforcing the required business rules.
- The social networking layer includes social listening, social analytics, social CRM, social governance, sentiment analysis, social marketing, social integration, blog/wiki, community, and knowledge base and provides seamless two-way access to social platforms from the digital channels, enabling enterprises to engage users actively.
- Content management is one of the main technology enablers for an enterprise digital platform. This module includes content authoring and presentation templates, content metadata, taxonomy, adaptive content, content authoring and publishing workflows, content security, content versioning, content backup and archival, content services, localization workflow, digital asset management, and integration with metadata management systems and translation management systems. The layer also handles other concerns such as migration, multi-site management, user-generated content (UGC), content administration, plug-in management, and content services. Document management modules manage digital documents and asset management modules manage the lifecycle of the digital assets.

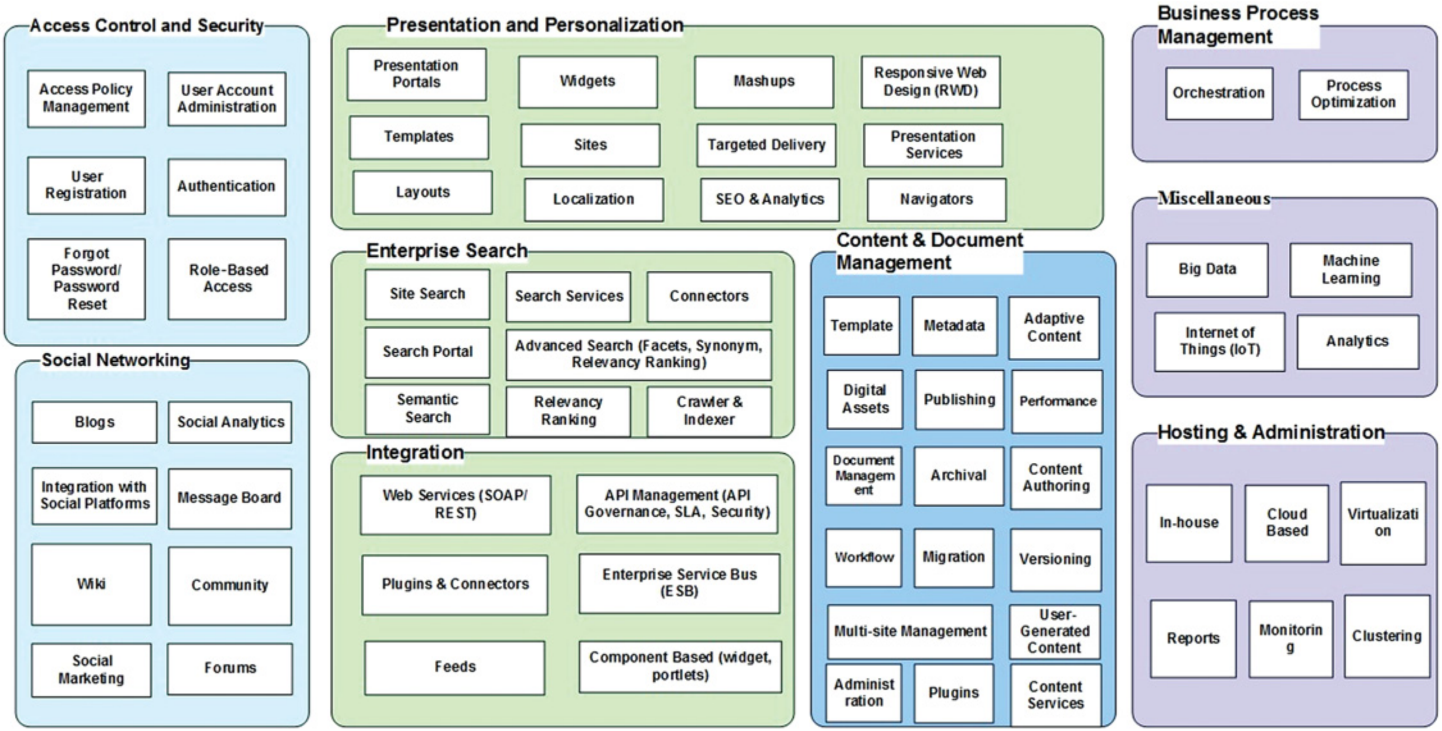


Figure 1.1 Enterprise Digital Capabilities

- Enterprise search is another predominant capability of digital platforms. Most modern digital platforms use search-centered experience to enable optimal information discovery and to facilitate self-service model. This includes search features such as site search, search portal, faceted search, synonym support, semantic search, and the like. Enterprise search system exposes its core features through search services for external systems. Plugins and connectors would be used to connect to various data sources for search indexing. Enterprise search also provides various configuration features such as relevancy ranking, business synonym configuration, artificial rank boosting, and such to enhance the effectiveness and relevance of search results.
- There are many emerging technologies such as big data, gamification, analytics, and Internet of things (IoT) and technologies such as machine learning and artificial intelligence that are also being used in digital platforms. These technologies help in analyzing huge data (such as social media, enterprise content, etc.), predictive analysis, natural language processing, intelligent search, and so on. B2C business enterprises would also include commerce capabilities such as registration forms modules, payment gateway modules, and other commerce modules.
- Hosting and administration capabilities include on-premise hosting, cloud, and virtualization technologies to provide on-demand content anywhere that lowers the operational and infrastructure cost. It also provides elastic scalability catering to growing business demands and agility. Administration also includes other infrastructure management activities such as health check dashboard reporting, infrastructure monitoring, migration, clustering, and others.
- The enterprise integration in modern digital enterprise is mainly based on services based on SOAP/REST and API with external/internal systems. Digital content is also exposed as feeds to external systems. An emerging trend in digital enterprises is that of API management. API management involves exposing business functions through APIs and integrating external systems through APIs to achieve cross-channel service virtualization using API Gateways. This category includes, among others, API governance, API security, API SLA compliance, and API analytics. There are other means of integrations such as component-based integration, connector/plugin/extension-based integration, and feeds-based integration. For complex enterprise integrations we use enterprise service bus (ESB) to provide a centralized integration and message exchange infrastructure.

The next section discusses how these digital capabilities enable specific business capabilities.

Digital Disruption across Business Domains

Earlier we had seen some of the exciting opportunities that digital capabilities have to offer. Here we extend that concept and examine how digital technologies can enable domain-specific features.

Figure 1.2 depicts functionality/features enabled by the digital technologies in various business domains.

Table 1.1 provides high-level summary view of changes across various business domains caused by digital technologies. This snapshot paints the picture of digital revolution that has redefined the business environment. Business capabilities in Figure 1.1 are mapped as business impact in Table 1.1.

As an example of digital transformation, we can look at the digital transformation of a health care domain. The digital transformation journey of a healthcare system consists of connected members and health care providers, digital wallet, digitized health records, Real Time Claims processing, Unified Enrollment & Payment, wearable device integration, marketplace model and such forward-looking features enabled by digital technologies. A typical digital healthcare platform consists of following features:

1. Commerce & Marketplace: Which consists of modules for Product Comparison & Recommendation, Cross Sell, Promotional campaigns on new plan offerings, Targeted campaigns based on current usage
2. Connected members: Unified view of Benefits, DocFind, Cost Share navigator, Preventive Care, Remote patient care, Notification and Care Alerts
3. Engaged provider: Clinical Data Integration, Access to personal health information, Real Time Claim processing, Referrals
4. Care delivery: Health Index Tracking, Treatment Options, Care Plan and Wellness adherence, Virtual Care – Video consultation, Tele medicine add-ons
5. Flexible payment modes: Advanced Payment Options – ApplePay, Google Wallet, Android Pay, Personalized statements
6. Loyalty discounts: Loyalty discounts, Discounts on early bird enrollment, Discounts on full payments, Adherence to Care Plans, Wellness Programs
7. Personalization: Personalization based on preferences, sentiment analysis, Recommendations on plans based on member history, conditions
8. Gifts: Sponsor insurance plan to loved ones, Sponsor part of premium or deductible
9. Discounts: Loyalty discounts, Discounts on early bird enrollment, Discounts on full payments, Adherence to Care Plans, Wellness Programs
10. Targeted Campaign: Promotional campaigns – free community health checkup, vaccination camps, Evidence based programs and practices – active parenting

In addition to the impact factors given in Table 1.1, digital technologies are also enhancing user's experience in other innovative ways. Here is a quick look at digitally enabled trends.

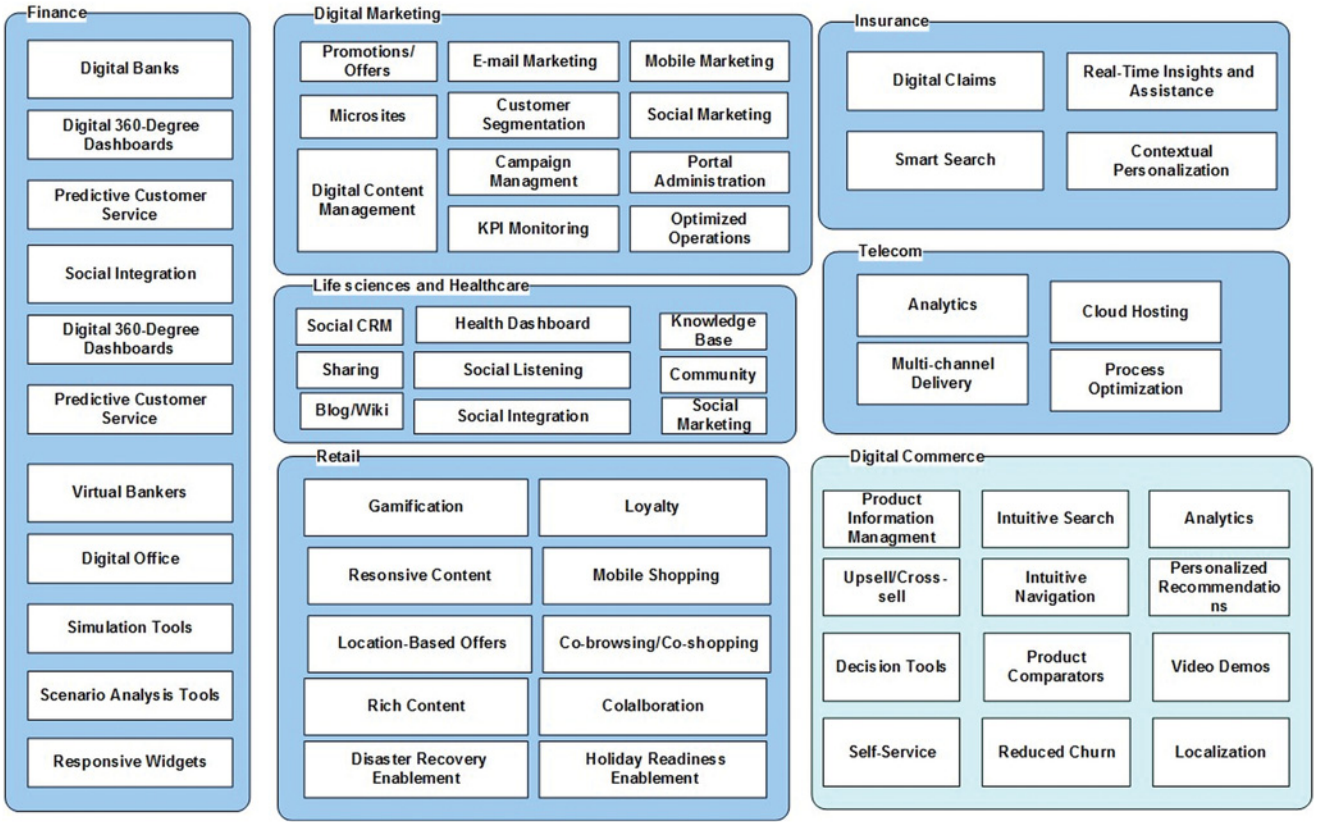


Figure 1.2 Business capabilities enabled by digital technologies

Table 1.1 Impact of digital technologies across business domains

Business Domain	Business Impact of Digital technology
Finance Industry	• Unified customer views through dashboards and 360-degree views • Real-time customer insights and Online Loan Application • Automated decision-making tools such as risk analysis tool, forecasting tools, financial calculators, advisory tools, budget planners, Portfolio Valuation tool, credit analysis, intuitive visualizations, and graphs • Digital branches, digital office, Personalized Investment Proposals, and virtual private bankers to provide a real-life bank branch experience to customers virtually • Predictive fraud and risk discovery tools and simulation tools to help customer take the informed decisions and aid in decision making
Insurance Domain	• Personalized offers and customer service to provide enhanced user experience • Enhanced scenario analysis, plan analysis, and plan comparison • Optimized claim processing through BPM and workflow optimization • Digital claims and smart search to promote a self-service model • Analytics-based targeted offers and personalized content delivery
Life Sciences and Health Care	• Unified health dashboard view of patients • Increased partnership with hospitals, pharma companies, and patients • Leverage the insights gathered from Internet-of-things (IoT) to get real-time health information • Integration with social media platforms to actively engage patients through social CRM, blogs/wiki, social marketing, Voice of customer (VOC), and virtual communities
Telecom Domain	• Cloud-based services to provide competitive plans and provide anytime-anywhere services • Analytics-based customer behavior prediction, churn calculation, predictive customer service, predictive offers/promotions • Multi-channel delivery on all customer access channels and devices.
Retail Domain	• Multi-channel, seamless experience through responsive and rich content • Mobile shopping through native or hybrid mobile apps • Optimized shopping experience through simplified checkout and order processing • Higher conversion rates, improved loyalty through personalized navigation and recommendations

Table 1.1 (*Continued*)

Business Domain	Business Impact of Digital technology
	• Increased cross-sell/upsell opportunities through personalized recommendations • Provided targeted content, advertisements and promotions, and personalized recommendations • Proactive problem resolution and personalized customer support • Context-aware offers and personalization based on location, time, geography, location, device, and other parameters • Faster times in launching marketing campaigns and microsites • Targeted campaign management by providing a holistic view of customer activities across various channels and using analytics-driven insights on customer behavior • More effective and cost-efficient digital marketing through real-time customer insights and personalization • Gamification of main business processes to provide effective incentives to customers • Development of holiday readiness strategies to handle increased demand and of disaster recovery plans to provide continuous availability • Influencing purchase decisions through immersive and inspirational content
Generic B2C Enterprises	• Engaging and relation-enhancing features • Incentivizing collaboration concepts and encouraging end-user driven co-creation. Leveraging gamification concepts for active engagement. • Provide holistic single-stop-shop dashboard view of all customer activities • Provide optimal Omni-channel experience • Enhance information discovery and easy-to-use user experience and navigation • Analytics-based personalization to enhance customer loyalty • User-preferences-driven user interface customization • Reduction in customer churns using predictive analytics • Personalized customer service • Enabling collaborative and self-service capabilities such as knowledge management systems (KMS), solution repository, document repository, and media database
Generic B2B enterprises	• Optimized process integration • Optimized business processes through content workflows • Increased collaboration across all business stakeholders • Business-centric process redesign

(*continued*)

Table 1.1 (*Continued*)

Business Domain	Business Impact of Digital technology
	• Productivity-improving tools and features
Digital Commerce	• Managing product content efficiently through product information management • Intuitive product search based on keywords, metadata, and other relevant parameters. Providing various filters to find the most relevant and appropriate product. • Analytics tools to understand customer behavior, popular product downloads, exit ratio, etc. • Personalized recommendation to increase the cross-sell and upsell opportunities • Intuitive and decision-making tools such as product comparators, etc. • Enhanced self-service model through FAQ, process automation, self-help content/videos, collaboration tools, product and solution knowledge base, etc. • Help with product information management (PIM) • Product review, sharing and rating features • Personalized promotions, offers, and loyalty programs to incentivize customer contribution

Emerging trends in digital platforms

Across business domains the key trends noticed in modern digital platforms are:

- Increasing popularity of human natural gesture interfaces such as touch-enabled smartphones, gaming consoles
- Responsive, interactive user interfaces that offer immersive experience
- Single-page applications (SPA) with simple and easy navigation model
- Social, mobile, analytics, and cloud enablement
- Active user engagement through collaboration, co-creation, and co-invention
- Hyper-personalized contextual content, service, and functionality.

Having looked at opportunities, challenges, and capabilities enabled by digital technologies, we now turn our attention to the key focus areas of this book: content management and search. It is evident from the above discussion that content management and search play a pivotal role in building a robust digital platform. Here we begin the journey of understanding these two key digital technologies in detail, starting with enterprise content management (ECM) concepts.

1.2 CONCEPTS OF ENTERPRISE CONTENT MANAGEMENT (ECM)

An enterprise content management (ECM) system manages all enterprise content, including Web content. For many functional domains, Web content management is also one of the concerns of ECM. ECM helps organizations handle the enterprise information efficiently:

- Integrated business processes linking inter- and intraorganizational boundaries lead to increased efficiency
- Provides quicker access to information when it matters the most
- Distributed scalable solution architecture is capable of handling anticipated growth
- Ease in meeting regulatory compliance for records keeping and handling
- Unified access to derived, personalized, and categorized information yields more business value
- Readiness for handling disaster recovery

In the next section we take a brief look at ECM concepts to understand the bigger picture and then elaborate on WCM concepts. Exploring key concepts of the ECM system at a high level helps us understand how the enterprise and Web content fits into the overall digital strategy.

Enterprise Content Ecosystem

Enterprise content management (ECM) is a solution platform that enables people to collaboratively create, manage, deliver, and archive information that drives business operations. ECM has evolved as a comprehensive solution platform for efficiently managing a wide variety of enterprise data. ECM involves various technology solutions to business problems associated with the production, storage, and distribution of enterprise information.

Enterprise content management (ECM) consists of managing the end-to-end lifecycle of all structured/unstructured enterprise content. Enterprise content is of diverse nature: it includes, among other elements, Web content, enterprise records, print content, data forms, enterprise assets, and electronic documents. Enterprise content forms the lifeblood of any enterprise. Enterprise information could be stored in variety of formats (structured and unstructured) such as Web content, digital assets (images, video, scanned forms), office document (e.g., PDF, MS Word), database data, collaborative content (e.g., blog, wiki, message boards, chat), corporate records, and rich media content.

Due to the diverse nature of enterprise content management, it encompasses several functions such as Web content management, digital asset management (DAM), workflow management, records management, Omni-channel management, search, collaboration, security, and others. ECM also deals with other enterprise content aspects such as, among others, content syndication, content migration, e-mail/collaboration content, paper and electronic content, metadata management, and localization.

The main features of an ECM system are as follows:

- Document Management: This includes secure authoring, indexing, versioning, and presentation/publishing of technical and/or legal documents. It includes management of all types of documents (paper, forms, proposals) throughout their lifecycle.
- Workflow and Business Process Management: This includes management and modeling of complex business processes such as document tracking, document and asset approval flows, and other business processes.
- Imaging: Consolidated storage and access of data-intensive static images and print stream data. Also includes forms capture and processing (OCR/ICR) and scanning.
- Web Content Management: Manage authoring, management, publishing, and presentation of text and/or graphical content on various delivery platforms.
- Enterprise records management to maintain enterprise digital records and forms.
- Digital rights management (DRM) to secure enterprise assets and content.
- Digital asset management to manage digital assets such as images, media, video, and other binary files.
- Enterprise search to index and organize all enterprise sources.

ECM also handles other capabilities such as storage, knowledge management, information lifecycle management, collaboration, case management, and the like.

A high-level view of ECM capabilities is shown in Figure 1.3.

The following is a brief overview of various capabilities depicted in Figure 1.3.

Content Presentation

The content access and presentation layer covers user applications, portals, and other business applications (such as business intelligence applications, reporting applications, dashboard portals, etc.). Online Web applications are other key user presentation components in this layer. Presentation APIs and presentation Web-services are used for integration across presentation applications and with underlying platforms such as document management, workflow, WCM, and KM.

Content Applications

Content management applications form the core of the ECM functional architecture covering applications around document and records management, workflow, WCM (web content management), enterprise search, content analytics, knowledge

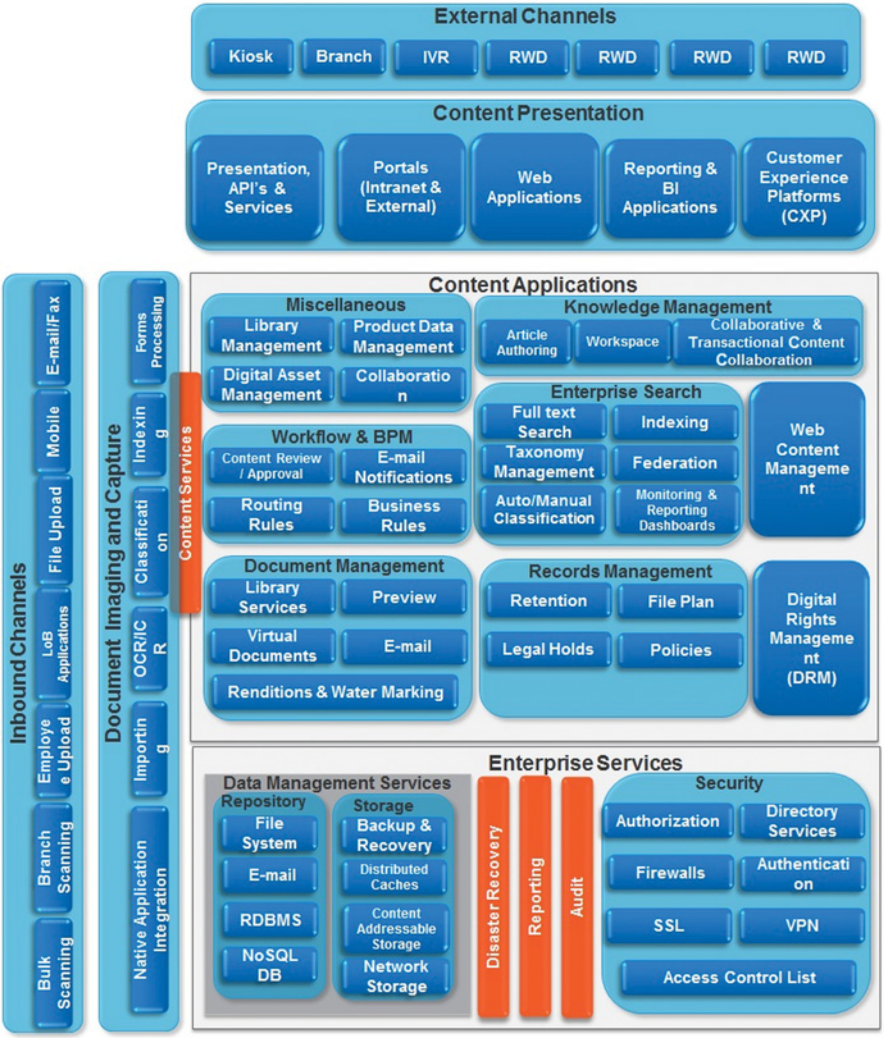


Figure 1.3 ECM Capabilities

management, and collaboration applications. These applications can be delivered by a combination of various technology platforms, with any single technology platform delivering capabilities within more than one functional area.

- A document management application provides document-processing capabilities including document library management, document preview, document rendition, document watermarking, and so on.
- Records management platform helps organizations be compliant to industry and legal regulations through enforcement and implementation of policies (such as retention policy, legal policy, disposition policy, etc.).

- ECM workflows support robust and complex processes and are closely modeled on business processes and implement related business rules. Business process management (BPM) capability is realized through workflows.
- A Web content management platform manages digital Web content and related assets. It provides management of complete lifecycle of Web content.
- Enterprise search and content analytics capabilities organize the enterprise data and use taxonomy to provide crucial business insights to aid decision making.
- Knowledge management modules provide knowledge creation (through article authoring tools), knowledge storage, and knowledge distribution. The modules are closely integrated with enterprise search to discover relevant knowledge assets.
- Digital rights management (DRM) module enforces access restrictions to the enterprise resources.

Other capabilities in this category are digital asset management (DAM), library management, collaboration, and product data management.

Enterprise Services

Services mainly fall into data management services and security services categories. Data management services include repository services, which abstract data sources such as file system, e-mail, and database; and storage services, which provide backup, network storage, recovery, distributed cache, and similar features. Security services include services related to authentication, authorization, and access control list, among others.

Access Channels

ECM platform could be accessed by a number of inbound channels (such as scanners, file uploaders, internal applications, etc.) and outbound channels (such as branch, kiosk, IVR, mobile, Web application, e-mail SMS, etc.). Enterprise documents are captured/ingested through OCR scanners, imports, integration, and file transformations.

The main focus of this book is on digital content management using Web content management systems (WCM). As we can see from the previous sections, Web content and digital assets are also some of the key concerns of ECM. Along with digital content management, we will also discuss underlying technologies such as workflows, digital asset management (DAM), document management, content analytics, content SEO, and others. In Part III we will also look at enterprise search, which is another key aspect of information management.

Note: Core ECM topics such as records management, business process management, digital rights management, and imaging are beyond the scope of this book.

Table 1.2 ECM vs. WCM

	ECM	WCM
Information type handled	Wide variety of structured and unstructured enterprise data such as documents, enterprise records, files, media files, records	WCM system mainly handles digital content such as Web content, digital assets
Workflow	Robust and sophisticated workflows to manage business processes Used mainly to model business process management (BPM)	Workflows mainly for handling digital content and assets
Publishing and content delivery	Publishing to a variety of platforms such as desktop, collaboration platforms, print media, enterprise systems, etc.	Publishing is mainly to Web and mobile platforms and as services and feeds
Applicability and usage scenario	Mainly for managing entire enterprise content	Mainly for managing online channels
Web capability	Mainly uses traditional Web channels and thick clients	Offers highly intuitive, responsive, and engaging Web experiences through Web 2.0 features
Target audience	Mainly internal enterprise audience	Internal and external web users
Social and collaboration	Features are implemented through integration with collaboration platforms	Provides robust social and collaboration features

ECM vs WCM

There are lot of overlapping concerns and features between ECM and WCM, but there are also subtle differences. Table 1.2 presents the salient features of both systems, which should allow us to differentiate their usage and applicability scenarios.

Technologies such as ECM, WCM, CMS, and portals handle overlapping concerns. Some of these technologies are also converging. For instance, a few portals are bundled with WCM capabilities and a few CMS systems also provide portal features.

Note: WCM is sometimes referred to as a content management system (CMS). However, there are differences between CMS and WCM. For instance, some CMS are designed to handle records, whereas WCM does not normally do records management.

As the terms CMS, WCM, and ECM are widely used in various contexts, the following section will help us understand better the exact scope of them, covered in this book.

Book's Focus Areas

Figure 1.4 depicts the features that are covered in this volume. Book scope items in content areas appear in bold. Topics such as workflows, asset management, and

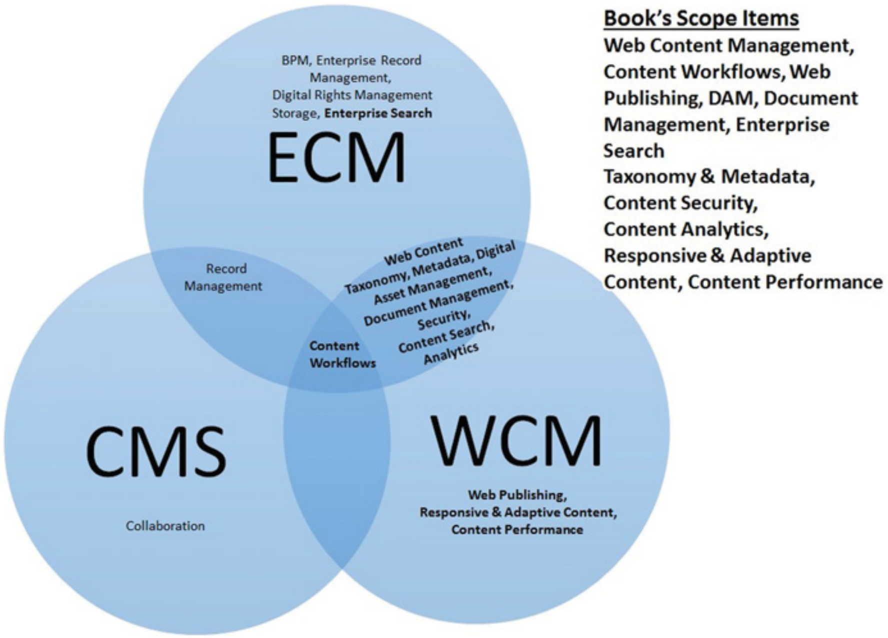


Figure 1.4 Book Focus Areas

Web content are at the intersection points of ECM, WCM, and CMS. Thus, the book covers all concepts of WCM, most concepts for CMS (except for record management and collaboration), and search and DAM topics in ECM.

Note: In this book CMS and digital content management are used to elaborate the content scope items depicted in Figure 1.3. Records management and collaboration are not covered as part of this book.

Digital technologies alone cannot ensure the success of a program. There should be a solid strategy within which the digital technologies would be put to use. A digital enterprise needs to have a robust strategy to take maximum advantage of any technology. Strategy provides the vision and direction, and ultimately leads to a comprehensive execution plan. In the next section we discuss the various aspects of enterprise digital strategy and content strategy.

1.3 ENTERPRISE DIGITAL STRATEGY AND CONTENT STRATEGY

We have seen various opportunities and disruptions caused by digital technologies to various business functions and domains. An enterprise embarking on a journey of digital transformation needs to develop enterprise digital and content strategies for a successful digital transformation. In this section we look at the key elements of these strategies. As the content strategy is a quintessential element for successful content management, Chapter 2 is dedicated to a more detailed discussion of it.

Enterprise Digital Strategy

Enterprise digital strategy defines the overall digital vision for an enterprise. It provides the direction for digital transformation and digitization of enterprise functions. It encompasses various other strategies such as, among others, content strategy, integration strategy, and user experience strategy. The digital vision lays out a road map for digital capabilities such as content management, portal, search, API management, and service-oriented integration to provide anytime-anywhere content.

A comprehensive three-dimensional enterprise digital strategy is depicted in Figure 1.5.

Figure 1.5 looks at the enterprise digital strategy from three perspectives. At the intersection of these perspectives we have depicted the digital enablers, processes, and digital technologies.

- **Digital transformation path** ranges from traditional (such as AS-IS scenarios) to personalized digital experience to responsive user experience to Omni-channel-enabled platform to a self-service platform. This is depicted at the top of Figure 1.5.
- **Key digital capabilities (on Y-axis)** are customer touch-point optimizations, integrated experience, and information value optimization. These capabilities act as functional enablers for the realization of a digital vision.
- **Core digital technologies (on X-axis)** are the technology enablers for the realization of digital capabilities. They include enterprise portals, content management system, enterprise search, and analytics.

Note: Figure 1.5 maps only the core digital capabilities and technologies and is mainly focused on portals, content, and search. Digital landscape is evolving and changing to cater to user needs. Hence there are other technology capabilities (such as IoT, AI, Big Data) that are not the focus areas of this book are not covered in Figure 1.5.

Let us look at the digital strategy elements in detail. We will first look at key digital capabilities on the y-axis:

Customer touch-point optimization: The focus should be on customer centricity for providing engaging relationships. We should analyze all customer touch-points across customer journey and optimize them to move from transaction-oriented to relationship-enhancing features. This includes personalized user experience, user-centric design, unification and consolidation of information, optimizing customer service functions, self-service capabilities, loyalty programs, personalized offers/campaigns, collaboration features, knowledge management, leveraging social platforms, and so on.

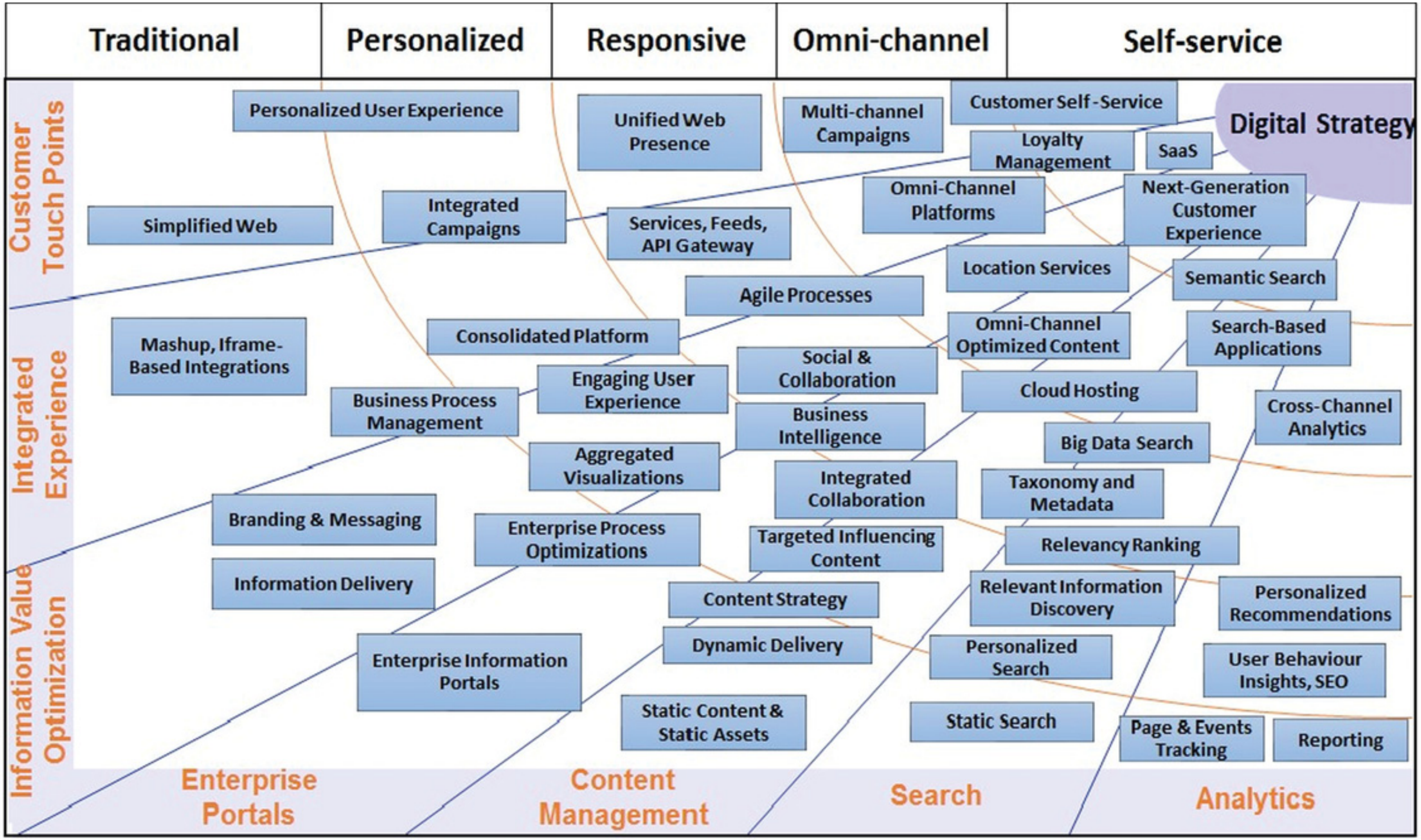


Figure 1.5 Enterprise Digital Strategy

Integrated experience: The digital platform should provide seamless and optimal experience on all devices to create an Omni-channel-enabled consolidated platform. Service-oriented architecture (SOA), API gateway, ESB, feeds, mashups, components, and widgets are the modes of integration for creating the next-generation customer experience that encompasses these capabilities:

- Personalized and relevant content delivery: This can be done leveraging analytics and implicit and explicit customer preferences.
- Optimal content strategy: This includes providing immersive, rich, engaging, inspirational content that is easy to find and easy to use.
- Digitally enabled legacy systems: Modernized legacy systems through wrapper services or migration initiatives.
- Digitally enabled business functions: Aligned business functions to leverage the digital capabilities. For instance, e-commerce platform can provide cross-channel integrated sales/promotion offers by leveraging analytics-driven customer insights and enhance cross-sell/upsell and conversion opportunities.
- Analytics: Gaining 360-degree insights into customer activities across various channels and using them to enhance user experience, provide personalization features, provide recommendations, and enhance a navigation model.
- Newer business models: Based on digital technologies, we can provide newer business models such as newer product delivery model (such as software-as-service [SaaS] model), newer revenue models (such as usage-based pay), newer hosting model (such as cloud-based hosting), and so forth.
- Business process management and process optimization: A fresh look at all business processes to understand optimization opportunities such as, among others, process automation and process optimization. Creating optimal process orchestration through process redesign and efficient workflow and rules management.
- Agile processes to become a responsive digital enterprise: Using an iterative delivery model allows the incremental buildup of the capabilities with faster turnaround times.

Information value optimization: Digital strategy should be able to use enterprise information as an asset to maximize its impact effectively. We need to move from static information delivery mode to integrated and engaging user experience. In order to achieve this, we need to consolidate, communicate, and conserve the information:

- **Information consolidation:** Information assets from various enterprise and external sources should be consolidated. This process includes content categorization, content processing, content modeling, intuitive information architecture, and similar activities. It includes the following:
    - Information strategy: Assessing information gaps and needs and defining ways to provide seamless information aligned with business goals.

  - Information architecture: Providing a blueprint and structure for information access.
  - Information integration: Consolidating information across various sources.
- **Information communication:** This includes providing the meaningful content to the target audience. Based on insights drawn from customer behavior, the content is optimized to suit user's preference and delivered in most optimal way on end-user's device. Optimal messaging strategy, branding, and integrated visualizations are part of effective information communication. It includes the following activities:
  - Information Delivery and access to enable and reach various devices and channels. Use intuitive visualization to convey differentiated experience.
  - Enterprise Intelligence to makes sense of enterprise data
  - Content Management to manage the Web content and related assets and documents
  - Real-Time Technologies for providing on-demand information
  - Intelligent Process Automation to deliver seamless integrated information
- **Information conservation:** This includes content evolution and adaptation to changing business scenarios. It includes the following activities:
  - Information Life Cycle Management to capture, store, revise, retrieve, preserve, destruct, and distribute content and data
  - Compliance and Risk/Security to protect information assets and be compliant with regulatory requirements
  - Governance to enable functioning of the overall strategy and information architecture. It also measures and monitors the effectiveness of the strategy and provides course correction actions when needed.
  - Capitalize on Information Model to leverage business value and drive business transformation

Let us now look at core digital technologies on the x-axis to enable digital capabilities.

Core Digital Technologies

Enterprise portals provide robust presentation and personalization capabilities in delivering feature-rich, integrated, engaging user experience. Content management system organizes enterprise content and assets through optimized content processes. The combination of CMS and presentation engine redefines the digital experience through targeted content delivery, integrated collaboration, and providing Omni-channel optimized content. Enterprise search plays a vital role in information discovery through flexible relevancy parameters. Enterprise search also provides advanced information retrieval capabilities through Big Data search and semantic search, and it forms the backbone for search-based applications. Analytics provides key user insights and helps in recommendations, metrics tracking, and reporting.

As we have seen, content strategy is one of the main ingredients of the overall enterprise digital strategy. It is also closely aligned with content focus area of this book.

Digital Content Strategy

Content strategy is the quintessential factor for the success of the content. In this section we provide a high-level overview of content strategy.

Content strategy is a comprehensive exercise concerned with creation, management and delivery of usable content aligned with an enterprise's business vision. Content strategy encompasses, among other factors, content planning, content delivery, content governance, and information architecture. It also leverages various related technologies such as digital asset management, workflow management, user experience, and others. As this is the core concept of content design, we have dedicated Chapter 2 to discussing content strategy in more detail. In this section we lay the foundation for content strategy and connect the elements related to content strategy with the book chapters.

Figure 1.6 provides various aspects of content strategy framework. This is a high-level framework that covers major aspects of content strategy used in the majority of scenarios. The content strategy framework covers the scope items, process, and content governance. Book chapters in Parts I and II are closely modeled on this content strategy framework.

Business functions requiring content strategy

Content strategy is required in almost all scenarios where content plays a major role. Digital marketing, brand management, information management, campaign management, Web site design, promotion management, online learning and education systems, and public relations management are some of the prominent business domains where content strategy plays a major role.

Content strategy should be fine-tuned and aligned with business vision. Business vision varies across domains. For instance *providing personalized and influencing content* forms the main business goal for an e-commerce domain. *Presenting easily accessible content* would be the primary goal of a health care domain; similarly, *engaging and interactive content* would be important for social and collaboration platforms, and *providing actionable content* is essential for a business intelligence domain. In all such scenarios the content strategy should aim toward providing usable, compelling, engaging, and actionable content that can fulfill the primary business goals. Content strategy helps organizations realize the key business goals such as:

- Unified marketing and sales experience with consistent branding across channels
- Consistent cross-channel experience
- Automated processes to reduce cost and increase productivity
- Social listening and sentiment analysis.

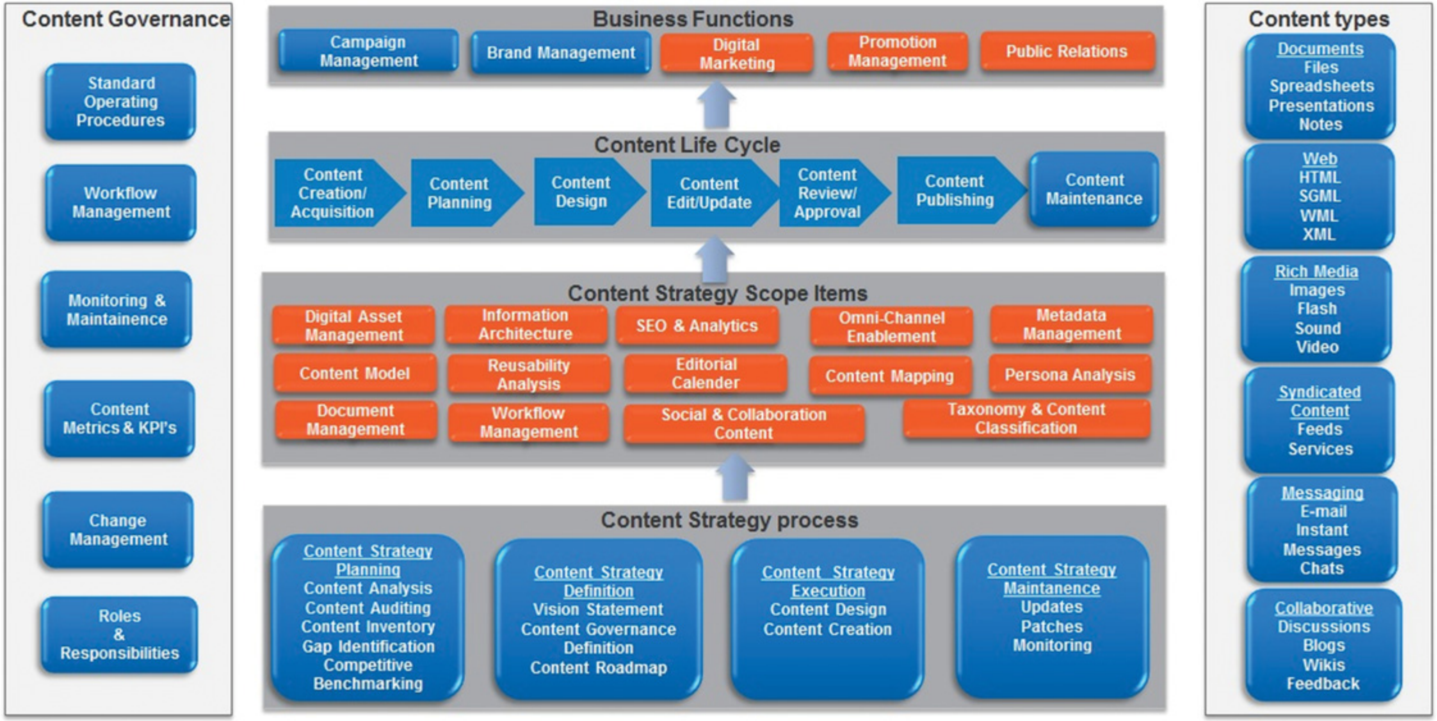


Figure 1.6 Content Strategy Framework

Content lifecycle

The principal aim of content strategy is to create and maintain meaningful content for the business. This naturally requires management of all content lifecycle phases starting from content acquisition through to maintenance and purging. During content strategy execution phase, a content strategist oversees content design and content creation activities including authoring content, editing, publishing, and managing content.

During content design and development phases, we define the content tone, style, messaging guidelines, and chunking strategy aligned with business goals. During content editing, we identify the frequency of updates, delta updates, and other similar aspects. For a more in-depth discussion, see Chapters 3 and 4.

Content strategy scope items

Content strategy covers the following areas:

- Digital Asset Management (DAM): All digital assets such as media files, graphics, and style sheets need to be properly managed. We will discuss this in detail in Chapter 8.
- Information Architecture: This includes aspects related to content discoverability, content navigation, content hierarchy, content and site structure, and other similar aspects. We will discuss this in Chapter 6.
- SEO and Analytics: SEO helps in bringing visibility to content, and analytics provides usage statistics for the content. We will discuss this in Chapter 10.
- Omni-channel enablement: This includes aspects related to rendering optimal content on all devices. We will discuss Omni-channel content (through adaptive technique) in more detail in Chapter 2.
- Metadata Management: Semantic metadata tagging helps in enhancing the content usage, discovery, and effectiveness. We will discuss this in Chapter 6.
- Workflow Management: Content workflows are closely modeled on business processes. This includes, among others, workflows related to content authoring, content editing, content publishing, and content localization. We will discuss this in Chapter 5.
- Content Security: This mainly deals with securing content and assets, which is discussed in Chapter 11.
- Social and Collaboration Management: User-generated content (UGC) forms a major part of social and collaboration platforms. This includes management of collaboration and social content (such as blog content, knowledge article content, wiki content, etc.).
- Taxonomy and Content Classification: This deals with categorizing and organization of the content. We will discuss this in Chapter 6.
- Content model that provides the structure for the content is discussed in Chapter 2.

- Reusability analysis to identify reusability opportunities is discussed along with content strategy in Chapter 2.
- Editorial calendar to provide the activities and road map for content activities is discussed in Chapter 2.
- Content mapping that maps the content to user journey steps is discussed in Chapter 2.
- Persona analysis is to identify the user segment and analyze the needs, wants, and goals of a particular user group. It is discussed in detail in Chapter 2.

Content strategy process

Content strategy definition is a multi-step process. This includes content strategy planning, content strategy definition, content strategy execution, and content strategy maintenance. Each of these phases involves various activities. We have discussed these processes in detail in Chapter 2.

Content types

Content strategy applies to various content types such as, among others, digital assets, documents, collaborative content, Web content, syndicated content, rich media content, and messaging content.

Content governance

Governance provides structured processes and well-defined role-responsibility matrix for smooth content operations. As part of content governance, we also will track content effectiveness through well-defined metrics and KPIs, and handle ongoing maintenance and change management activities. Governance is covered in Chapters 2 and 17.

After having looked at ECM big picture along with enterprise digital and content strategy, it is now time, in the next section, to discuss digital content management and search.

1.4 DIGITAL CONTENT MANAGEMENT AND ENTERPRISE SEARCH: AN OVERVIEW

Digital Content Management

From the previous discussion we have seen that content management is one of the key components of a digital ecosystem. Digital content management involves end-to-end managing the content lifecycle starting from design and ending with maintenance. It provides features for, among other things, metadata tagging, content workflows, content publishing, content analytics, and content translation.

Value addition of CMS to digital platform

CMS is one of the main digital capabilities that organizations need. CMS provides the following features:

- Provides a centralized content management platform providing a single view of content of enterprise data
- Streamlines enterprise content-related processes to make them more efficient and agile
- Helps enterprises stay competitive through faster turnover and efficient publishing
- Provides content in a structured format so that it can be reused across various applications and services.

Enterprise Search

An enterprise has content distributed across various data sources in various formats. For instance, information about an employee could be distributed across employee database, finance ERP, and internal collaboration platform. In order to get meaningful data about an employee, it is essential to organize the information holistically and present it in relevant context.

Enterprise search plays this key role of organizing and integrating the enterprise information and making it easily discoverable in the relevant context. Digital enterprises can reap various benefits from search technologies. Search function can be positioned as the key navigation enabler and information discovery tool in a prominent position on the page. As this is one of the main focus areas of this book, Part III is dedicated to a detailed discussion of the enterprise search technology.

Value addition of enterprise search to digital platform

Search enables digital platforms to achieve multiple business goals such as:

- Relevant information discovery through relevant search results
- Dynamic content aggregation of all enterprise content sources
- Promotion of self-service models through solution search, saved search, etc.
- Cross-selling and upselling of products by bundling related products through product search and recommendations
- Promoting newly released products through techniques such as artificial relevancy boosting
- Exposing the enterprise information to third-party systems through search services
- Assisting in legacy modernization and digital transformation by indexing the legacy content and making it available through search functions.

Enterprise Sources Used by Search Engine

Normally an enterprise search engine uses these enterprise content sources to index and organize information:

- Online Web content: Search engine can crawl and index all enterprise Web applications using the base/start URL.
- Secure content: Search engines also support various authentication mechanisms to crawl and index secured systems (such as private Web pages). The search results for authenticated content are made accessible for authorized roles.
- Network file share content: The search engine crawler can access content/files residing in file servers.
- Enterprise databases: Search engines use database connectors to index data from a database.
- ERP systems: Search engines use services or adaptors to index data in ERP systems.
- Services: Search engines can also consume (and expose) content from internal or external Web services.

1.5 CHAPTER SUMMARY

- In this chapter we discussed the high-level concepts of enterprise content management system (ECM).
- ECM system provides end-to-end management, storage, retrieval, and publishing of enterprise data in various formats.
- Features of ECM include document management, workflow management, imaging, Web content management, records management, digital rights management, digital asset management, enterprise search, and collaboration.
- The chapter discussed differences between ECM and WCM and presented the book's focus areas.
- The chapter presented various enterprise opportunities and capabilities that can be realized by digital technologies. Digital technologies provide various opportunities such as Omni-channel enablement, social engagement and collaboration, real-time insights, personalized delivery, self-service capabilities, productivity improvements, streamlined processes, compliance adherence, content consolidation, and optimized administration.
- Digital trends and challenges were discussed. The main challenges are in the area of enterprise integrations, process optimization, meeting market and consumer expectations, collaboration, consolidation, and culture change.
- Enterprise digital capabilities include, among others, capabilities related to user experience, social engagement, access control and security, content

management, enterprise search, process management, integration and administration, and API management.

- Digital technologies enable various business functions such as finance, digital marketing, insurance, telecom, digital commerce, retail, life sciences, and health care
- Enterprise digital strategy should include digital vision, customer centricity, seamless Omni-channel experience, information consolidation, information communication, and information conservation.
- Content strategy is needed for efficiently designing and managing content.
- Digital content strategy includes managing content lifecycle for an optimal experience.
- Digital strategy includes various elements such as DAM, document management, information architecture, SEO, analytics, metadata management, workflow management, social engagement and collaboration, and taxonomy.
- Content strategy includes various processes such as, among others, content analysis, competitive analysis, content visual definition, content design, and reusability analysis.
- Digital content management involves end-to-end management of digital content throughout its lifecycle.
- Enterprise search helps in organizing enterprise information and improves it findability.
- Enterprise search adds value to an enterprise through relevant information discovery, dynamic content aggregation, self-service, content/product promotion, and search services.
- Enterprise search indexes data from sources such as Web content, secured content, file sources, enterprise database, and ERP systems and services.

Chapter 2

Content Strategy

Content is at the heart of content management. Content conveys information, engages users, and inspires, motivates, and influences end user behavior. A robust content strategy is the quintessential ingredient for the success of enterprise content. We have conducted a general overview of content strategy in Chapter 1. In this chapter we will have an in-depth look.

First we need to understand the main goals and attributes of good content, and we begin our discussion with this. We also examine common challenges associated with enterprise content. In the subsequent sections we explore the essential prerequisites for defining a robust content strategy. Here we will see the content best practices, content requirements, elaboration and other content analysis activities. In the next section we look at four phases of content strategy along with key design considerations. We will have a detailed look at various activities in each of the content strategy phases. The section that follows elaborates on key elements of content strategy such as personal analysis, content model, content mapping, reusability strategy, chunking strategy, adaptive content, editorial calendar, and effective content design. The chapter concludes with a content strategy case study in which we will apply the content strategy concepts to e-commerce content.

Content architects, content strategists, CMS developers, and enterprise architects will find this chapter useful.

2.1 OVERVIEW OF CONTENT STRATEGY

In this section we explore the primary goals of a content strategy exercise, as well as related strategies. We also look at common challenges we face during the content strategy exercise. The section lays the foundation for the rest of the chapter contents. In the coming sections we look at various methodologies and content strategy activities to address these challenges and fulfill the content strategy goals.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

Introduction to Content Strategy

Content strategy is all about planning for optimal usage of content to serve the business needs. It includes various governing principles, processes, and best practices that cover the entire lifespan of content from its creation all the way to archiving and purging. Its ultimate goal is to make the content *usable* to end-users while fulfilling organization goals. Content should engage the end-user by providing useful information and helping them take action on the basis of informed decision. It establishes clear-cut content governance processes for optimal maintenance and usage of content. Content strategy involves various aspects of content such as compiling, auditing and optimizing various content types and content sources, aligning information architecture, using metadata for optimal content delivery, modeling data, modeling navigation, creating workflows, and defining content governance processes. To deliver content effectively, an organization or entity must have a robust content strategy based on solid design principles.

The key stakeholders of content strategy are:

- **Business stakeholders**: They are mainly interested in using content strategy for increasing conversion rate and sales revenue and achieving business goals. Content strategy should be able to create the content that influences the end-user while effectively communicating the brand values using relevant voice, tone, and message.
- **Marketing stakeholders**: They are interested in the search engine optimization (SEO) to drive the user traffic. Content analytics, campaigns management, and promotion management are other areas of interest for these stakeholders. Content strategy should address these concerns while making provisions for rich and engaging interactive content.
- **User experience designers**: This set of stakeholders includes creative agencies and in-house User Interface (UI) engineering and design teams. Content strategy should make provisions for the design and development of forward-looking User Experience (UX) elements to provide optimal experience for the end-users.
- **Content stakeholders**: This set of stakeholders (including content writers, reviewers, and publishers) are interested in the content processes related to authoring, publishing, translation, and other governance aspects related to content.

Program managers, information architects, and integration architects are other stakeholders who are interested in activities of content strategy. End-users and consumers naturally are the key benefactors of a robust content strategy.

Main Tenets of Content Strategy

The primary tenets of enterprise content strategy are listed below. The content strategist has to work with all related subject matter experts (SMEs) and teams to define a robust content strategy.

- Content strategy defines the *basic tenets* of content:
  - How content is presented (such as tone, voice, look and feel, etc.)
  - Where content resides (such as systems, channels, site areas, etc.)
  - What content consists of (such as messaging, structure, content story)
  - When content becomes available (such as update frequency, publishing frequency, syndication frequency)
  - Who the intended users are (such as target audience, personalized delivery)
  - How content is used (such as content services, feeds, sharing, collaborative content, user-generated content)
  - What the content's impact on the user is (such as inspirational content, influencing content, effective call-to-action content, self-service content)
  - How content is aligned to business goals (using content metrics and KPIs, governance model)
  - How content is searched and found (using metadata, information architecture, taxonomy, content model)

Content strategy should aim to optimize basic content tenets.

- Increase the *efficiency and consisten*cy of the content to make the content usable. This includes communicating the brand message in consistent and uniform ways across all delivery channels.
- *Optimize the content* for each of the delivery channels (such as Web site, mobile devices, micro-sites, blogs, newsletters, collaborative content, shareable content, video, Web cast, documents, reports) to make it more contextual, relevant, personalized, localized, and impactful for the target audience.
- Design the content *to meet the business goals* such as increase in sales conversions, user engagement, user retention, user loyalty, content collaboration, and such. It also articulates how the proposed content changes will help in meeting business goals.
- Define *content governance* processes for end-to-end content management. It establishes processes, tools, standards, and policies for the entire content lifecycle, and it maps the owners for each of the governance steps.
- Establish a *continuous improvement* process for sustained content maintenance.
- *Optimize various supporting elements* of content ecosystems such as information architecture, metadata, SEO, analytics, and social engagement and collaboration.
- *Communicate consistent* brand message to the target audience through optimal messaging strategy and persuasive communication strategy to increase brand credibility

Related Strategies and Artifacts Used in Content Strategy

Content strategy has broad scope and it overlaps with other strategies and concerns. Content strategy is closely associated with following strategies:

- Information architecture
- Communication strategy
- Metadata strategy
- Content editorial strategy
- Content execution strategy
- Workflow design
- Template design
- SEO and analytics design
- Content governance

A content strategist plays the role of primary contributor, reviewer, and overseer of the activities listed above. Chapter 6 is dedicated to the discussion of metadata and information architecture, Chapter 5 to discussing templates and workflow, and Chapter 10 to discussing SEO and analytics.

The following list contains the main artifacts of content strategy exercise. We will discuss these in more detail in the upcoming sections. A content strategist would play the role of reviewer during delivery of some of these artifacts.

- Content gap analysis report, which identifies the gaps in the existing content ecosystem.
- Copy deck, which provides sample content as an authoring aid for authors. The primary responsibility for this artifact is with the content design team.
- Content testing plan, which provides detailed scenarios and cases for testing content under various conditions. The primary responsibility for this artifact is with the content validation team.
- Wireframes and mockups, used to create content templates and structure the content. The primary responsibility for this artifact is with the content design team.
- User journey and persona modeling, used for optimizing the user experience and fine-tuning the processes.
- Omni-channel plan, used to delivery content optimally on all channels.
- Metadata and tagging plan, used to create and tag the metadata for content. Content strategist collaborates with information architect in the development of this artifact.
- Taxonomy plan, used to create the site taxonomy. Content strategist collaborates with information architect/taxonomist in the development of this artifact.
- Workflow analysis and optimization plan, which documents all the existing content workflows and recommends improvements.
- Content auditing reports, used to analyze and benchmark existing content.
- SEO and analytics plan, used to integrate the SEO and Web analytics with content.

- Standards and guidelines definition plan, which includes defining Web standards, accessibility standards, content integration standards, content service standards, UI standards, JavaScript standards, etc.
- Social engagement and collaboration plan, used to create and maintain collaborative content.
- Industry benchmarking report, used to benchmark existing content and processes against industry standards and best practices.
- Competitive benchmarking report, which evaluates existing content and processes against competitors.
- A visual style guideline, which defines the user interface standards and specifications for UI elements (such as layout, font, color, spacing, graphics, UI elements, etc.) and delivers consistency and uniformity. The primary responsibility for this artifact is with content design team. Content and editorial style guide govern content creation and help in effective content management.
- Content authoring and presentation template, used for authoring and presenting content. Content strategist collaborates with content architects in the development of this artifact.
- Editorial calendar for depicting the timelines of content-related events such as campaigns, publishing milestones, etc.

Common Challenges with Content

Determining the challenges commonly associated with content ecosystems should help us define best practices and techniques to address these challenges in content strategy. This section presents an overview of these challenges, along with best practices to alleviate them; both are addressed in more detail in the later sections.

- **Duplicate content across Web site sections, pages, and channels:** While there are multiple reasons for the occurrence of duplicate content, the main reason is the lack of proper content governance process and absence of reusability strategy. Comprehensive content governance and enterprise content consolidation are needed to address this.
- **Content silos**: This mainly occurs due to absence of centralized content ecosystem and lack of collaboration among content creators. This is also the common cause of duplicate content, because the involved parties are more likely to duplicate their efforts without proper collaboration. Effective communication and collaboration and content governance are needed to avoid content silos.
- **Inconsistent and non-uniform content:** In some scenarios, each business unit has unit-specific UI standards, page structures, and style guides. Ultimately when all those pages are rendered on the same Web site, user experience is affected. This leads to inconsistent navigation, poses challenges in finding relevant information, and does not convey the message or brand value properly.

An enterprise-wide visual style guide and navigation model should be defined for addressing this.

- **Complex and inefficient content processes**: While modeling the business processes, content workflows tend to become complex, requiring numerous steps and multi-level approvals. This further impacts the content publishing timelines. A well-designed and optimized workflow strategy is required to understand the pain points and optimize the workflow wherever possible. All duplicate, redundant workflow steps can be removed and the approval steps can be automated and made time-bound wherever possible.
- **Less effective content leading to decreased usability and conversion rate:** This could include large-size monolithic content without proper metadata, labels, and tags, making content hard to find, read, and understand. A/B testing and multivariate testing can identify the optimal usage and positioning of content and assets.
- **Content discovery challenges**: Information discovery is one of the main challenges. Content may be hidden behind multiple levels, and users may face challenges accessing it. Effective metadata strategy, intuitive information architecture, logical content categorization, friendlier navigation models, and metadata-based search techniques would help make content discovery easier and faster.
- **Slower turnaround times**: Too much of time required for content authoring and publishing could delay the time-to-market. One of the ways to address this is to optimize content flows and content reusability strategy. Flexible and reusable authoring templates along with WYSIWYG editors, seamless integration with DAM systems, document repositories, metadata systems, and Web analytics systems would optimize content authoring and publishing times.
- **Content integration challenges:** Upstream and downstream systems find it difficult to consume content. In some scenarios the external systems directly connect to content repositories and employ nonstandard interfaces for the integration. This design is not scalable and is prone to defects during CMS upgrades. In order to address this, we need to use standard content service interfaces (such as REST/SOAP-based content services) to expose content. XML or JSON format can be used for content services.
- **Omni-channel delivery:** Various versions of content are created to cater to various delivery channels such as mobile, desktop browser, and so on. This leads to lot of duplicate content that is difficult to manage. It is necessary to separate core content from presentation and to use different presentation templates to format the content for various channels. It is necessary to serve DITA/XML-based content or JSON-based core content for various systems, which can add presentation logic as per their needs. It is most effective to use adaptive content to address this issue.
- **Challenges with content collaboration:** Content is not easily shareable across various users, channels, and social media platform. Often content is duplicated

because of this. An effective content collaborative strategy is needed to address this challenge. Also, the content presentation platform should provide content-sharing ability features across internal and external platforms.

In addition to these challenges, there could be other issues such as absence of content reusability strategy, challenges with communication strategy, absence of enterprise-wide taxonomy and metadata, and lack of content governance. In coming sections we take a look at various ways to address these challenges through content strategy.

Definition of Key Terms

Before we discuss details of content strategy, let us look at the key terms used throughout this chapter for better clarity.

- **Metadata:** It brings meaning and context to the content and is tagged with the content. It makes content discovery easier.
- **Content messaging:** It communicates the broad information and ideas to the end-users. It includes, among other things, appropriate labels for elements, usage of standard terminology, and consistent message flow.
- **Information architecture:** It defines the design and organization of information in the delivery channel. It includes navigation model, site maps, and site taxonomy and is mainly aimed at improving content findability.
- **User persona:** Persona models a given user type or user group. It includes user attributes, user demographics (such as age, education), user goals, and user needs and expectations.
- **Taxonomy:** Enterprise taxonomy defines, stores and manages a hierarchy of terms and their relationships. Taxonomy terms could be used for content tagging, content classification, and content organization.

2.2 PREREQUISITES FOR CONTENT STRATEGY

Content strategy is a broad term and has to be optimized based on the business domain and problem context. While we can define core elements of the strategy that can be applied to all domains, it is equally important to fine-tune the strategy to a specific business scenario to work well for the organization's needs. Before we start discussing the elements of content strategy, we will lay the foundation for content strategy. In this section we take a look at the key essentials and building blocks of content strategy. These essential elements would serve as supporting aids for the core elements of content strategy. They would also be prerequisites that provide inputs for building a robust content strategy.

The key prerequisite for content strategy is to develop comprehensive content requirements covering various aspects of enterprise content. A complete requirement is essential in content analysis and content design. The next section discusses various aspects of content requirements.

Content Requirements

Thorough understanding of the content needs from various dimensions from all concerned stakeholders (such as end-users, authors, and business) is essential for building an effective content strategy. Popular customer channels, most frequently visited content, and popular content formats are some of the areas that provide insights into customer needs and expectations. A content strategist will gather insights about end-user experience through feedback mechanisms such as user interviews, Web analytics, surveys, beta testing, multivariate testing, and so forth. During user research and user interviews, a content strategist explores the existing content ecosystem and processes, role-responsibility matrix, content ownership, current gaps and pain points, existing tools, and various content formats, among other things. User research is conducted through surveys, focused group discussions, workshops, data analysis, and user interviews. This helps identify improvement opportunities, user preferences, and gaps and helps with forecasting and decision making.

The key activities in requirements elaboration (from end-user standpoint) are as follows:

- Analyze the voice, tone, and message that have maximum impact on customers. We look for consistency in the content message.
- Understand various customer segments and personas and analyze the content requirements, interests, and goals for each user segment. Map the needs, interests, expectations, and decision influencers for each persona. This can be achieved by modeling the user journey for each user persona. We are going to discuss persona analysis as part of content strategy discussion.
- Get the business inputs from various stakeholders through interviews and joint workshops.
- Analyze the digital assets requirements that are used along with content. The digital asset format (image, video, binary file), asset positioning on the page, and asset size need to be analyzed.
- Understand language and geospecific requirements. This includes the need for translated content, language-specific text alignment, etc.
- Identify the site areas and content sections that are more frequently visited by site users. Most frequently visited pages, popular content, highly downloaded documents, and frequent paths all need to be carefully analyzed. Any gaps or issues in these areas may have a drastic impact on user experience.
- Analyze the requirements for decision-making content to influence user behavior. Aggregated views of related content, product comparison content, and unique selling point (USP) for product content are some instances of influencers.
- Analyze content workflows related to content creation, content review, content localization, and content publishing.

- Gather insights about customer interaction and behavior from Web analytics. We need to understand the bounce rate – pages/content sections where customers are exiting – and improvise them.
- Compile customer feedback from various channels such as surveys and polls.

All these points help us in understanding critical content touch points that matter to customers most. Other dimensions of the content requirements are discussed below.

Content NFR requirements analysis

Compile all the nonfunctional requirements (NFR) related to content such as content security, content performance, content accessibility, content scalability, content usability, content availability, content modularity (leading to reusability), and others.

Omni-channel content requirements analysis

Analyze the channels and devices from where content is accessed. Based on the business domain, target audience, and use cases, the access channels may vary. For instance, tech-savvy audiences mainly use mobile channels and are interested in social and collaboration features. Traditional B2B content is normally accessed through desktop browsers. Based on the channel requirements, we need to design the content optimized for those channels.

Content analytics analysis

Web analytics provide vital insights into customer behavior and content usage. The metrics will help us understand the exit rate, frequent downloads, time spent on site, and so on. We could use the metrics to identify the areas of improvement.

Social and collaboration analysis

Compile and analyze all social and collaboration content needs and gaps for active customer engagement. This could include things such as missing social media integration and lack of collaboration features such as blogs, wiki, community, etc.

Business-related requirements

In addition to this, we need to carefully analyze the content needs for critical business scenarios from a business standpoint. Content strategists analyze the content requirements for all the business use cases. User journey modeling and persona identification help identify the critical steps and content needs in each of the stages in the user's journey. For instance, in a typical e-commerce domain, a purchase scenario goes through three stages. Content needs for these stages are as follows.

- **Research Phase:** The content in this phase includes content on home page, product information pages, product detail pages, knowledge base content, blog content, solution article content, search results content, etc. The primary intent of the content in this phase is to convey complete and accurate product information in an effective way and highlight the compelling differentiators to draw user's attention. Rich media text, digital assets, interactive and usable content, and collaborative content are used widely in this phase. By the end of this phase, user would have gathered more information about the topic of interest.

- **Decision-Making Phase:** Users explore more data points and content to perform the intended task (such as finding the exact solution article, finalizing the product, etc.). This stage requires content that can help customers make informed decisions. This includes content such as product comparison information, product feedback content, content star rating, and the like.
- **Action Phase**: In this stage the user performs the intended action. This may be something like purchasing the product, downloading the document, or completing a workflow. The content and workflows in this stage should be optimized so that user can complete the action quickly.

In all these stages, the content usage metrics are measured so that they can be used for optimizing the content.

So far we have seen the description of content requirements that form the main input for a content strategy exercise. Once we compile comprehensive content requirements, we can initiate the content strategy definition process.

2.3 DEFINING CONTENT STRATEGY

Various analysis steps and essential elements defined in previous section provide an overview of improvement areas and provide inputs for defining a robust content strategy. This provides a good foundation for us to define the content strategy in detail. In the following section we take an in-depth look into content strategy, including various phases, attributes, design considerations, and output of content strategy.

Phases of Content Strategy

Content strategy is concerned with various aspects of content along with its ecosystem elements. Broadly we can define four phases of content strategy:

- Content strategy planning: In this phase the content strategist mainly analyzes existing content and understands business goals through input from various stakeholders.
- Content strategy definition: The content strategist defines the overall vision and creates a road map for the enterprise content. The content strategist also articulates how the vision addresses current gaps and fulfills intended business goals.

- Content strategy execution: This phase involves all aspects of content design, with the content strategist overseeing the content evolution throughout its lifecycle.
- Content strategy maintenance: This is an iterative phase in which content is tracked based on its metrics and the learnings are used for future content optimization.

Key activities, deliverables, and objectives of content strategy phases are depicted in Figure 2.1.

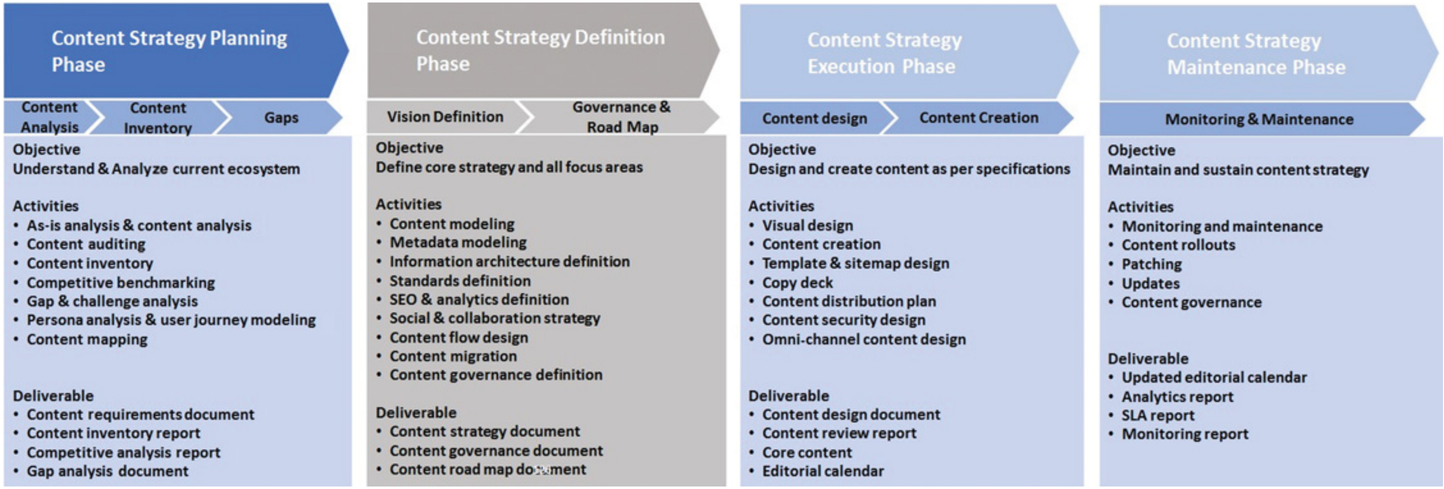
Content strategy planning

In the planning phase, the main focus is on understanding and analyzing the current content state. A content strategist interviews business stakeholders and conducts user research to obtain insights about the business goals and end-user expectations. The main activities in this phase are as follows:

Content Analysis: Analyzing the enterprise content forms the foundation for future content strategy. Various types of enterprise content are analyzed in this step. This includes analysis of all structured and unstructured enterprise content, digital assets, content workflows, publishing flows, content volume, content analytics, and SEO, among others. Analysis of existing content tools and all content sources is also necessary. A content strategist in this phase also closely looks at existing content ecosystem. Understanding of content ecosystem elements such as content repository, DAM system, document repository, and security system can help the content strategist in identifying opportunities for reusing, retiring, or replacing existing systems. During this stage, the content strategist also looks at the current content ownership (such as product content owned by an e-commerce system, Web content owned by a CMS system, etc.). Competitive analysis and benchmarking are also among the key activities in this phase. The content strategist interviews all stakeholders to understand business objectives. We have seen some of this content analysis as part of "Understanding content requirements" in the previous section. The output from this phase includes identification of areas of improvement, content audit report, analysis of existing content processes, identification of roles and owners for each of the process steps, compilation of branding and messaging gaps, flow diagrams, and business rules documentation. Based on the analysis of how content is used, it is also possible to identify all duplicate, unnecessary, outdated, inaccurate, and incomplete content. Content usage analysis also helps in future design decisions. For instance, if the home page content needs to be displayed on both desktop browsers and mobile devices, the templates can be designed accordingly.

Content Auditing: Content audit is one of the key steps in content strategy planning. Auditing mainly includes identifying existing content sources, content types, and formats. A typical content audit document consists of following information:

- Content information: unique content identifiers for the content, content format, content sources, and content languages



Content Strategy Planning Phase

Content Analysis › Content Inventory › Gaps

Objective
Understand & Analyze current ecosystem

Activities
- As-is analysis & content analysis
- Content auditing
- Content inventory
- Competitive benchmarking
- Gap & challenge analysis
- Persona analysis & user journey modeling
- Content mapping

Deliverable
- Content requirements document
- Content inventory report
- Competitive analysis report
- Gap analysis document

Content Strategy Definition Phase

Vision Definition › Governance & Road Map

Objective
Define core strategy and all focus areas

Activities
- Content modeling
- Metadata modeling
- Information architecture definition
- Standards definition
- SEO & analytics definition
- Social & collaboration strategy
- Content flow design
- Content migration
- Content governance definition

Deliverable
- Content strategy document
- Content governance document
- Content road map document

Content Strategy Execution Phase

Content design › Content Creation

Objective
Design and create content as per specifications

Activities
- Visual design
- Content creation
- Template & sitemap design
- Copy deck
- Content distribution plan
- Content security design
- Omni-channel content design

Deliverable
- Content design document
- Content review report
- Core content
- Editorial calendar

Content Strategy Maintenance Phase

Monitoring & Maintenance

Objective
Maintain and sustain content strategy

Activities
- Monitoring and maintenance
- Content rollouts
- Patching
- Updates
- Content governance

Deliverable
- Updated editorial calendar
- Analytics report
- SLA report
- Monitoring report

Figure 2.1 Content Strategy Phases

- Content discoverability: content metadata, tags, social tags, information architecture, and navigation model
- Content measurability: content analytics, content KPIs and tracking metrics, and user satisfaction index
- Content business value: business metrics, brand messaging, marketing and campaign effectiveness, and promotion management
- User impact: content usability, interactiveness, decision influencers (content related to call to action, share, buy, influence, decision tools, compare, form submit, register), and content accessibility
- Content gaps: duplicate content, out-of-date content, content performance issues, and content process gaps

Content health assessment: Catalogs all assets and artifacts needed for content strategy and assesses its quality. This includes, among others, structured and unstructured content, metadata, digital assets, workflows, visual style guides, branding elements, templates, taxonomy, content migration processes, and localized content. It helps with identification of reusable and obsolete content and enables content migration, content assessment, and content strategy exercises. The content health assessment creates content inventory and provides us with the understanding of content location and content structure. The inventory is useful for content migration and the creation of a content road map.

Persona analysis and user journey mapping: Identifies key user personas by profiling target audiences. Maps the needs, wants, expectations, and other details for each persona and models their journey while on the site. Personas help design content efficiently and deliver personalized and targeted content. As this is a key step in understanding content needs, we are going to discuss this in detail as part of content strategy elements discussion.

Gap identification: Identifies all the gaps for the items identified in content inventory, as well as the ones related to content usability, accessibility, reusability, performance, and other features. A content strategist also looks closely at issues that prevent existing content from achieving the business goals (e.g., What is causing users to move away from content? What are the reasons for low site traffic?). The content strategist also performs content usability assessment and competitive analysis and conducts benchmarking against industry trends and best practices to identify any gaps within existing content.

Competitive analysis and benchmarking: Competitive analysis includes comprehensive comparison of Web, content, and processes compared to direct competitors. It benchmarks the existing content, current content standards, and prevalent content processes against the competition and industry standards, best practices, and customer expectations. In this step, a content strategist performs critical analysis of existing content state and benchmarks content against industry standards, organization brand and design guidelines, competition, and end-user needs and expectations. Benchmarking can be done on various parameters such as content usability, branding, content structure, information architecture, content performance, interactiveness, and so on. This analysis provides clear areas for improvement and identifies gaps, constraints, and key business processes that can be included as part of content strategy. It enables stakeholder decision making and helps them take course-correcting measures to compete better.

Current challenge analysis: Looks critically at current paint points and identifies gaps in the existing content framework. This may be in the content processes, content usability, information architecture, metadata strategy, or other areas. The output from a gap analysis exercise provides a list of opportunities and areas for improvement.

In an overview, the content strategy–planning phase is an extension of the requirements elaboration phase with the main focus on content analysis. The output from the planning phase includes content requirements document, content inventory report, content audit report, competitive analysis report, content benchmarking report, and gap analysis document.

Content strategy definition

The main content strategy is defined in this phase. Strategy definition involves defining core strategy, content governance, and content road map.

Defining the core strategy: In this step the content strategist defines the overall vision and lays out the focus themes. Content strategy is a broad area and involves various elements:

- Content model based on the content structure requirements of the site: Content templates use the content models. Selection and customization of CMS to support the templates is planned during this step. These will be discussed in greater detail in the coming sections.
- User-centric content design: Content should be easy to use and should contain consistent voice and tone, visual and style guides, themes, branding guidelines, and messaging guidelines to convey the uniform brand identity. A content strategist collaborates with creative agency and UI designers for this. This also will be discussed in detail in the coming sections.
- Metadata and taxonomy: Based on insights on content consumption, a content strategist can work with a taxonomist to define the metadata strategy. Chapter 6 is dedicated to discussing this.
- Information architecture: A content strategist plays a key role in defining scalable information architecture from the content standpoint. This includes, among others, navigation design, URL design, site structure design, and sitemap.
- Content standards: A content strategist must define all content standards needed for content creation, content integration, and content sharing. We will discuss various content standards in Chapter 7.
- SEO and content analytics guidelines: This includes defining SEO checklist (such a keyword, description, image alt tags, page meta tags, title) that can be used while content authoring and editing. Additionally, the authoring template

can also provide placeholders for SEO tags. Content-tracking code and metrics/KPIs to monitor the content usage should also be defined. Chapter 10 is dedicated to discussing this.

- Social and collaboration content design: A content strategist is involved in design of social and collaboration strategy, which involves content syndication, content sharing, and content services.
- Content flow design: Based on existing gaps and benchmarking data, a content strategist comes up with optimal flows for content processes.
- Omni-channel content design: Based on analysis of target audience and end-user devices, the content design has to be optimized to work well on all target platforms. This involves adaptive content design. We are going to have a detailed look at this in coming sections.
- [Optional] Content migration: A content strategist is involved in migration design, selection of migration tools, planning automated and batch migration, planning migration phases, migration testing, and other similar activities.
- Knowledge base creation: As a content strategist has vital insights into content design, gaps, best practices, and content flow, this knowledge has to be documented in a centralized knowledge repository. This helps the extended team use the subject matter expertise to build on the existing design. Content checklist, design guidelines, and content best practices also can be stored in this centralized repository. This knowledge base can further be used for training new team members.
- A content strategist puts together a content design checklist for content requirements, content creation, content aggregation, and content migration. The checklist may also contain best practices for designing and developing content. This serves as a gating criteria and validation tool.

Defining content governance: Content governance processes provide a management framework for content operations and provide well-defined content roles and responsibilities. We have discussed content governance as one of the key elements of content strategy in this chapter.

Defining content road map: A content strategist defines the road map to realize the business goals. This road map lays out the foundation for future release rollouts, content updates, editorial calendar, and content milestones. A sample content road map is given in Table 2.1

Design Considerations

The key content design considerations while defining content strategy are as follows.

- Content findability: We need to consider the factors to enhance the discoverability of the content. This includes, among others, information architecture, content navigation models, content metadata, social tagging, content search, taxonomy, intuitive site structure.

Table 2.1 Content Road Map

Business Goals	Quarter 1	Quarter 2	Quarter 3
Content consolidation	Audit various enterprise content Design for content integration	Migrate the content into single content platform	Add social and collaboration features Add content syndication feature
Consistent brand identity	Identify gaps in existing UI elements Create contemporary and forward-looking visual style guides, UI standards, Omni-channel guidelines, and messaging guidelines	Design the consistent content based on the standards and visual style guide Define the governance processes for content updates	Adhere to content governance processes for regular content rollouts Iteratively test the content on all supported channels
Content process optimization	Identify gaps and challenges with existing content processes Define content governance	Redesign and optimize content processes related to workflow, authoring, and publishing	Design and implement self-service features into content processes Enhance content discoverability Automate steps in content approval process

- Content editorial process: We need to understand the existing editorial process and gaps and challenges with current process. This can be used to optimize the editorial process.
- Content approval process: We should understand the steps and business rules, process steps, and levels of approvals needed in the current content approval process. Content strategists can then try to optimize the process.
- Content-tracking metrics: We should also design the metrics and KPIs that can be used to track the usage of the content. Content metrics can quantify the success of content design.
- Content reusability strategy: While designing a robust content strategy, one of the main focus areas is enhancing content reusability. We need to explore all avenues to reuse the content across various channels and devices. We have detailed the reusability strategy in coming sections.
- Content visual design: Visual styling and branding guidelines are among crucial design aspects of content strategy.

- Competitive benchmarking: During the content strategy exercise we should also benchmark the content, information architecture, and visual styles against the competition. We should also identify opportunities for using applicable industry standards.
- Content maintainability: Focus should also be made on establishing future maintenance and enhancement process going forward.
- Content governance: Overall content governance model should be established to manage and optimize content processes and to define the role-responsibility matrix.
- Continuous improvement: Content strategy should define the process for continuous content improvement. Even after content publishing, content usage should be continuously monitored through content analytics, user feedback, and by other means. This information should be used to continuously fine-tune content, SEO, structure, information architecture, metadata, etc.
- Social enablement: Content strategy should be able to support the social-enabled and user-generated content for active user engagement.
- Services and feeds: Content should be available through services and feeds for other systems. Services and feeds distribute the content, and users get constant updates through this subscription model. This is another way of active user engagement.

The key output from strategy definition phase is the content strategy document (consisting of focus areas such as content model, messaging, metadata, infrastructure architecture, analytics, SEO, Omni-channel, and visual style guide), content flowchart, content governance document, and content road map document.

Content strategy execution

Strategy execution primarily consists of content design and content creation activities. The content strategist ensures that content design and creation are consistent with the specified guidelines, best practices, and principles defined as part of previous content strategy phases. The content strategist will play a reviewer role, providing oversight of these activities.

Content design: In this step, design principles (defined as part of content strategy definition phase) are used for designing content structure. Content workflow design, content template design, content site map, content messaging design, content model design, wireframes, mockups, editorial calendar, content migration design, copywriting, asset design, content security design, content distribution plan, and content metadata design are included in this step. The next section takes a closer look at some of these activities. Omni-channel content design (through adaptive techniques) is also part of this step. Designing for content readability, accessibility, and usability is implemented in this design exercise.

Content creation: A content strategist oversees the content creation process to ensure that it adheres to defined standards and specified governance processes. The content strategist ensures that created content is consistent with defined branding guidelines, communicates the message with defined voice and tone, and ensures that content is optimized for all delivery channels. The content strategist closely reviews content creation, usage of templates/visual guidelines, style guides, usage of templates, and usage of copy deck for content authoring and asset creation. Content is created with specified SEO, accessibility, analytics, and Omni-channel specifications. CMS platforms are widely used in this phase.

Note: Normally wireframes are accompanied by copy decks, style guides, and visual designs. Style guides specify the standards of content and asset elements (such as font size, image specification, logo, navigation style,s etc.) Copy decks provide sample content for the wireframes, which guides the author. Authors can use both style guide and copy deck during content creation process for creating content with consistent branding style and format.

The key output from the strategy execution phase is the content design document, content review document, and core content (as XML/HTML or in other equivalent forms).

Content strategy maintenance

Regular ongoing content maintenance and content operations-related activities are part of this phase. Few activities in this phase require oversight from a content strategist.

- Content updates and patches: Periodic content updates, content testing, regular content audits, updates to content inventory, and content rollout based on content road map and editorial calendar are some of the key activities of this step.
- Continuous monitoring and maintenance: This includes fine-tuning the content messaging, content workflows, and SEO based on the continuous feedback received from end-users and business stakeholders. Web analytics is used to understand the usage and effectiveness of content through content metrics. Metrics and user feedback are employed to improve user adoption.
- Content auditing: Content is audited on a periodic basis to catalog the content and assets. During this exercise a content strategist identifies opportunities for content/asset reusability and analyzes any potential gaps.
- Content governance: The content governance processes handle the content update and change management processes. A content strategist oversees the adherence to defined governance processes and fine-tunes the processes based on the feedback.

The output from this phase includes content editorial calendar (for depicting content releases), reports related to content analytics, content performance, and content scalability. A content strategist analyzes the analytics reports to understand the effectiveness and fine-tunes the content flows, content design, and content metadata wherever necessary.

Core Elements of Content Strategy

We have seen some elements of content strategy in various content strategy phases. We will now explore the core elements in greater detail, as this may help in robust content strategy development. In this section we will look at persona analysis, content mapping, content model, content reusability, and adaptive content with the help of examples and case studies.

Persona analysis and user journey modeling

Identifying user personas and modeling their user journey is a key exercise in content requirements phase. *Simply put, a user persona is a representation of site users and reflects all attributes of a given user segment such as demographic details, needs, expectations, and interests. User personas categorize users into user groups to efficiently identify their concerns, interests, characteristics, likes, dislikes, usability areas, goals, and expectations.*

Personas help better design the content, information architecture, processes, and personalization features specifically targeted for each persona. Persona analysis helps in developing robust content strategy and provides inputs to other activities such as, among others, content mapping and user-centric content design. Content strategists can design highly engaging, inspiring, and persuasive content based on persona analyses to offer compelling user experience.

We start modeling user personas by understanding the audience and user segments. Each unique user segment with distinct needs, wants, expectations, motivations, and purchase intents may constitute a user persona. We can use formal feedback channels (such as surveys, feedback forms, user interviews, or focus workshops) and use analytics reports to gain insights into user behavior. Social listening, multivariate testing, and user profiling are other techniques to know about various user personas. We could collect various data points related to a user persona through these techniques. Data points include demographic details (role, language, geography, age), information access details (pages accessed, tools used, social channels, challenges, preferred content formats), access channel details (Web, mobile, tablet), expectations (performance expectation, interests, primary influencers, challenges), tasks (activities, tools used, process gaps), and goals. In an e-commerce application, user groups such as visitor, customer, reseller, site administrator, prospect, retail customer, corporate customer, and support executive are some of the distinct user personas.

Once we identify user personas, we map out user journeys for each of the user personas, so that we can clearly understand the information consumed and challenges faced in each step of the journey.

Let us look at sample user personas and a user journey to understand these concepts better. We take an example of an e-commerce site for this small case study. The main personas for an e-commerce site are visitor, retail buyer, wholesale buyer, and customer support person. We will employ user surveys – feedback analysis and analyze analytics reports to get more details about each these personas. We define the *retail buyer* persona as follows:

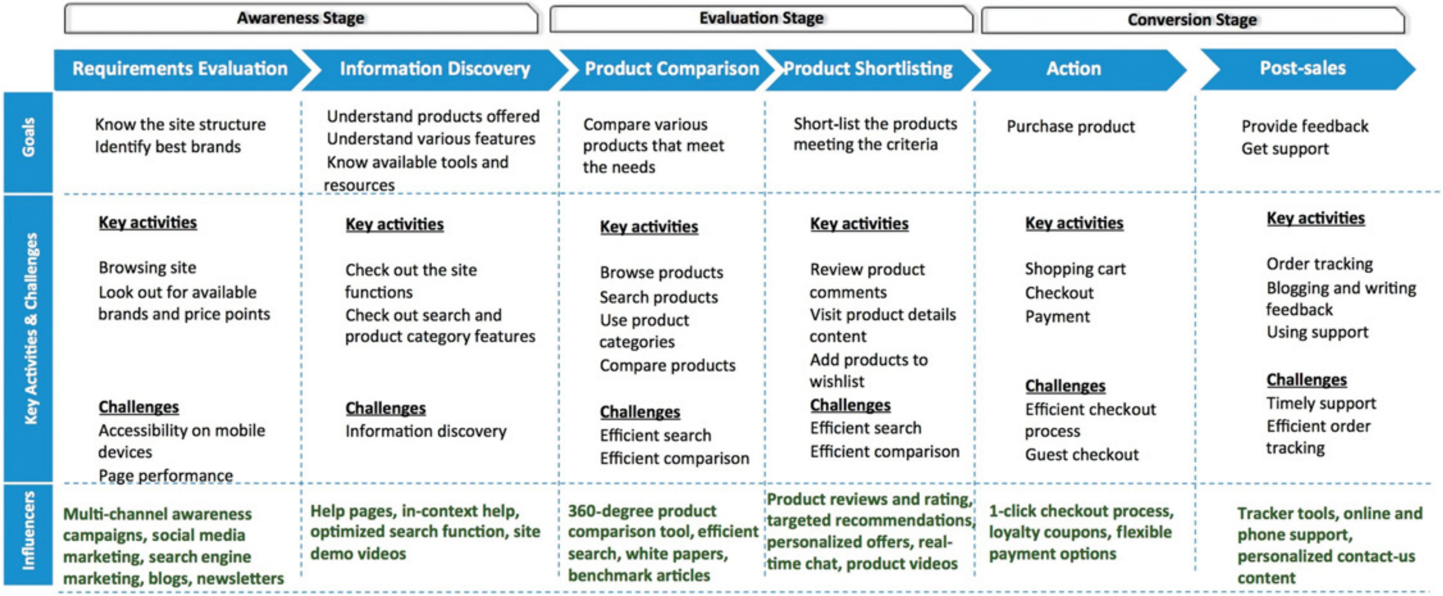
Retail Buyer Persona	
Location	United States
Age	20-30 years
Average Yearly Income	60k-90k USD
Language	English
Buying motivations and goals	Buy the products with best brand value with good after-sales support
Concerns	Product prices
Interests	Sports, fashion, networking, blogging
Channels	Mobile, tablet, and web
Buying influencers	Product comparators, personalized promotion offers, loyalty programs, product rating, social recommendations, review comments
Likes	Product review and rating, product forums and blogs
Most common activities/tasks on site	Search products, compare products, view products, review products, and purchase products
Challenges	Too many steps in shopping process, product search showing irrelevant search results sometimes
Tools and resources used	Product comparator, product specification document

Let us model the shopping journey of a retail buyer persona. We broadly classify the persona activities into three phases: awareness, evaluation, and conversion. Figure 2.2 provides a detailed picture of the purchase journey with goals, key activities, challenges, and influencers in each phase of the journey.

In the awareness stage the user evaluates the requirements (wants and needs) and discovers the information on the Web site. The user browses the site and uses product pages and a search tool for information discovery. Awareness campaigns, search tool, in-context help, SEO/SEM, demo videos, and social media marketing could be used as main tools to influence user's behavior in this phase.

During the evaluation stage, the user compares various products to find the product best suited for his/her needs. The user employs advanced search features and product comparison tools at this stage. Optimized search, benchmarked product reports, real-time chat, product videos, product comparison tools, product reviews, and personalized recommendations are key influencers and decision-aiding tools in this phase.

During the conversion phase, the user purchases the product and uses feedback channels to provide the feedback. User also tracks the order shipment and may use support at this stage. Efficient checkout experience, loyalty incentives, flexible payment options, personalized support and tracking tools enhance user's experience in this phase.



	Awareness Stage		Evaluation Stage		Conversion Stage	
	Requirements Evaluation	Information Discovery	Product Comparison	Product Shortlisting	Action	Post-sales
Goals	Know the site structure Identify best brands	Understand products offered Understand various features Know available tools and resources	Compare various products that meet the needs	Short-list the products meeting the criteria	Purchase product	Provide feedback Get support
Key Activities & Challenges	**Key activities** Browsing site Look out for available brands and price points **Challenges** Accessibility on mobile devices Page performance	**Key activities** Check out the site functions Check out search and product category features **Challenges** Information discovery	**Key activities** Browse products Search products Use product categories Compare products **Challenges** Efficient search Efficient comparison	**Key activities** Review product comments Visit product details content Add products to wishlist **Challenges** Efficient search Efficient comparison	**Key activities** Shopping cart Checkout Payment **Challenges** Efficient checkout process Guest checkout	**Key activities** Order tracking Blogging and writing feedback Using support **Challenges** Timely support Efficient order tracking
Influencers	**Multi-channel awareness campaigns, social media marketing, search engine marketing, blogs, newsletters**	**Help pages, in-context help, optimized search function, site demo videos**	**360-degree product comparison tool, efficient search, white papers, benchmark articles**	**Product reviews and rating, targeted recommendations, personalized offers, real-time chat, product videos**	**1-click checkout process, loyalty coupons, flexible payment options**	**Tracker tools, online and phone support, personalized contact-us content**

Figure 2.2 Retail buyer persona's purchase journey

Content mapping

Content mapping is an activity that maps enterprise content to fulfill user needs and achieve business goals. Content maps aim to answer various questions during user journey and organize information intuitively. Content mapping is another form of content inventory preparation by mapping content required for each stage in the user journey. We take all the identified user personas, model their journeys, and map the content for each step. During this process we also identify the content gaps. It helps in identifying existing content and the content that needs to be created to address concerns and needs of each persona.

The first step in the content-mapping process is to identify user personas and map the journey of each of those personas. We have seen the details of persona analysis and journey modeling in the previous section. For each step in the user journey (mainly in activities and influencers section), we map the existing content. We use the inputs from content inventory for this mapping. If there is no corresponding content or tools, we identify a "content gap." A content strategist comes up with design to address these gaps.

Let us create a content map for the retail buyer user journey we identified in the earlier section. Table 2.2 provides the content map for this.

From Table 2.2 we can identify all the required content types and existing content gaps. Required content types can be taken from existing content inventory, or content can be newly created if none exists. We can create/update content to fill the gaps identified in Table 2.2. We can perform content mapping for all personas for comprehensive content mapping.

Content models

The content model provides structure for the content and depicts the messaging value of the content. The content model defines all necessary content types along with their relationships and ensures that information is created and depicted with the specified structure consistently in all delivery channels. Content modeling is the starting point in the journey of structured content design. Content models can effectively translate the content requirements and user experience elements into structured and implementable content elements. Similar to a data model representing the database data, a content model represents the high-level content types along with their relationship hierarchy. Content models break down a large-sized, monolithic, unstructured content into flexible, modular, structured, and reusable content blocks. Through content models a content strategist can communicate the content requirements to content developers and CMS architects. Developers and CMS architects can use the content models to create the content templates, page layouts, information architecture, and navigation models and design efficient content interfaces.

Content-authoring templates or XML schemas, which can enforce the specified structure, are closely modeled based on the content model. Content models depict "level of details" conveyed by the content and relationship among content elements.

The main attributes of a content model are:

Table 2.2 Content map for purchase journey of retail buyer user persona

	Challenges	User Goals	Business Goals	Content Type Needed	Content Gaps
Requirements Evaluation Step	Omni-channel access Performance	Understand site structure	Spread awareness, educate users	Campaign content, blogs, e-mail, newsletter	Absence of adaptive content
Information Discovery Step	Efficient information discovery	Explore information	Provide right information at the right place	Home page content, help content, categorized content (new arrivals, best sellers)	Absence of rich-media content such as site demo videos
Product Comparison Step	Efficient search, efficient product comparison	Compare products	Provide decision-making tools, user engagement tools	Product comparison content, product white papers, case studies, user guide	Absence of product review content, absence of product forum content
Product Shortlisting Step	Efficient search, efficient product comparison	Shortlist products	Intuitive shopping experience	Product demo videos, quick view content, “bought together” content	
Action Step	Efficient checkout process	Purchase product	Increase conversion rate	Click-to-action images, quick checkout content	Absence of analytics-based conversion tracking
Post-sales Step	Efficient support	Get support and provide feedback	Increase loyalty and repeat customers, self-service	Contact-us content, feedback form content, FAQ content, return form	Absence of context-based contact-us content, personalized support

- **Definition of content type**: A content model should define all unique content types required for a given solution domain. For instance, in an e-commerce domain, various content types would be a short description of the product, product image, product video, product solution article, and the like.
- **Specification of content structure:** For each of the content types, the content model specifies the structure including, among other things, content elements, attributes, and sequence. For instance, a short description of the product would consist of content elements such as product title, product subheading, product description, and a thumbnail image of the product.
- **Specification of restrictions:** All restrictions to the content elements such as maximum word count, data type, data format, and language type should be specified by the content model.
- **Specification of relationships**: A content model also depicts interrelationships between various content types.

In some cases content types also specify the read-only fields (such as audit fields) and controlled vocabulary values (such as country list, workflow list)

The content model helps in packaging content for a solution domain. We can define content model at both page level and content chunk level.

Let us look at two content models in detail. The first content model is used for creating product Web content through the "*product_web_document*" content model. It consists of all attributes used while rendering the product content. All required attributes are obtained during the requirement elaboration phase of content strategy. Since the "*product_web_document*" content type represents the content structure of the product content, it provides all the necessary details and structure needed to model a product content. At a high level, the content model for this product covers the information in these categories:

- **Authoring fields:** These fields are depicted in the authoring templates and are mainly used by authors to enter the required information. Fields such as product_short_descr, product_long_desc, product_page_title, and product_seo_tags can be updated by the author in the authoring templates. The fields would specify the content type (such as content chunk, image, video, etc.), various product attributes (such as title, description), and enforce restriction (data type, maximum word size, etc.)
- **Audit fields:** These values are read-only values that are automatically populated by the system: Product_approver, Product_approval_date, Product_workflow_stage, Product_version, Last_version_date, and Archival_date are examples of this.
- **Controlled vocabulary values:** Some of the field values are obtained from a controlled vocabulary. Author/System can only select from a predefined set of values for these fields. Product_target_audience, product_product_type, product_product_family, product_product_name, product_target_site, and product_category are examples of these fields.

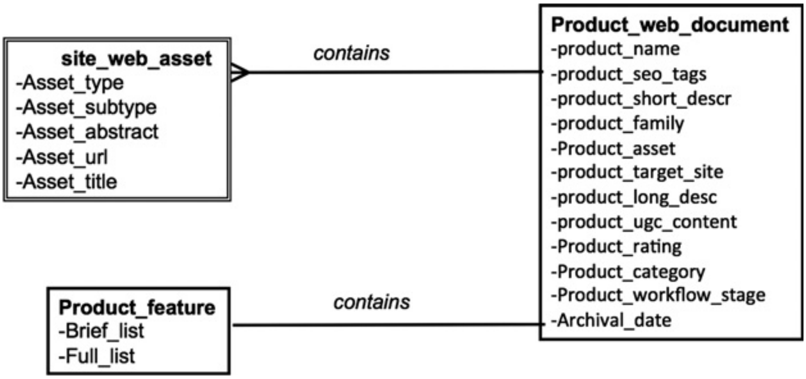


Figure 2.3 Sample Content Model for e-commerce content

- **End-user fields:** In Web 2.0 scenario, end-user also co-creates some of the content. In product content scenario, end-users can specify product_ugc_content (to store product reviews) and Product_rating.
- **Relationship field:** These fields provide relationship between content models. In a product_web_document content model, "product_asset" field would refer to "site_web_asset" content model for attaching a product image.

A simple content model for e-commerce content is depicted in Figure 2.3. It shows relationship between product content, product feature, and product image. Product content (content type = product_web_document) may contain multiple product images (content type = site_web_asset) and may contain a product_feature (content type = product_feature). Figure 2.3 shows the list of elements/attributes for each content type.

Now let us look at the details of *product_web_document* and *site_web_asset* content types.

In the Table 2.3 we saw a detailed content type for product content. Most of the Web content requires digital assets (such as images, videos, etc.). Let us look at a detailed content type for depicting a digital asset.

site_web_asset content type can be used in all places where assets are required. For instance, the *product_asset* of the *product_web_document* can use the *site_web_asset* content type to create a product image. The site_web_asset content type is shown in Table 2.4.

A sample mapping of the Web page elements to the attributes of site_web_asset content type is shown in Figure 2.4.

How do we create content models?

Content strategists must consider the following factors when creating a content model for a solution.

Unique Content Types: Content strategists must compile all possible content types based on inputs from content analysis exercise in the strategy-planning phase. Content analysis would provide an effective overview of the content structure.

Table 2.3 Product_web_content Content Type

Content Type	product_web_document				
Description	Parent type for all objects representing product web content				
Attribute Name	Display Name	Mandatory?	Data Type	Length	Description
product_seo_tags	SEO	N	String	100	All applicable SEO tags for the product (such as keywords, teasers, part names, etc.)
product_target_audience	Target Audience	N	String	64	This field provides the intended target audience (consumer, reseller, vendor)
product_short_descr	Product Brief Description	Y	String	200	Provides brief description of product content
product_family	Product Family	N	String	255	Family of the product
Product_asset	Product Image	Y	Ref (site_web_ asset)		Product image. This can use the site_web_asset content model
product_name	Product Name	N	String	255	Name of the product from product management system
product_target_site	Target Site	Y	String	32	The site where target content will be rendered
product_long_desc	Product Description	Y	String	400	Detailed description of the product
product_ugc_content	Review comments	Y	String	500	This field holds the user generated content (UGC) entered by end-users as part of review comments
Product_rating	Rating	Y	String	1	Provides the aggregated rating value (on a scale of 1 to 5) for a given product specified by the end users.
Product_category	Product category	N	String	50	The category of the product. Product content will be stored in the appropriate folder structure
Product_workflow_stage	Workflow Stage	N	String	10	Current workflow stage of the product content. Should be one of New, Ready for Review, Approved, Ready for publishing, published
Archival_date	Archived Date	Y	Time	50	Date when the content was last archived. Will be populated once the content is archived

Table 2.4 Site_web_asset Content Type

Content Type	site_web_asset				
Description	Object type for binary content destined for the Web (images, videos, PDFs, SWF, etc.)				
Display Label	Site Web Asset				
Name	Display Name	Req.	Data Type	Length	Description
asset_type	Asset Type	Y	String	100	Specifies the asset types (image, document, video)
asset_subtype	Asset Sub-type	Y	String	100	Specifies the asset subtypes
asset_abstract	Asset Abstract	N	String	500	Brief description of the asset
asset_url	Asset Link URL 1	N	String	128	Hyperlink for the asset (Repeatable)
asset_title	Title	Y	String	50	Asset title

Content-mapping activity based on persona analysis and user journey modeling also provides required content types. A content strategist can then identify unique content types from the overall list. For instance, in product content solution, content analysis may reveal various usages of product content (such as product description, product feed, product image, product short description, product long description, etc.) on product pages, and we can identify the unique content types as *product description* and *product image* (product feed, product short description, and product long description can be derived from product description content type). Additionally, the content

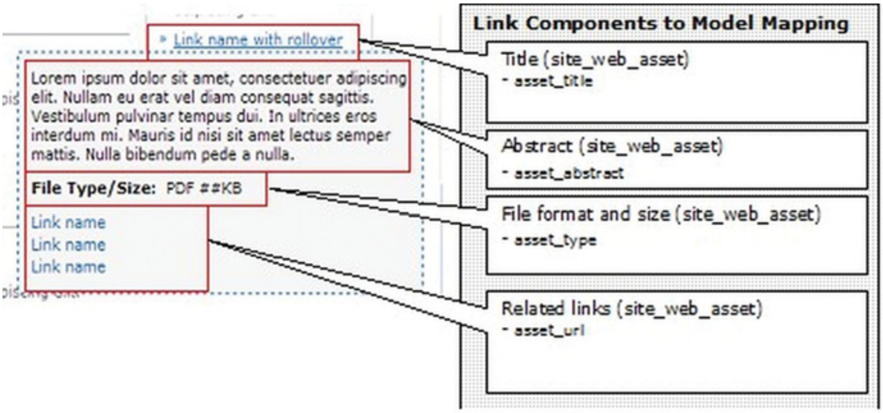

Link name with rollover
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nullam eu erat vel diam consequat sagittis. Vestibulum pulvinar tempus dui. In ultrices eros interdum mi. Mauris id nisi sit amet lectus semper mattis. Nulla bibendum pede a nulla.
File Type/Size: PDF ##KB
Link name
Link name
Link name
Link Components to Model Mapping
Title (site_web_asset)
- asset_title
Abstract (site_web_asset)
- asset_abstract
File format and size (site_web_asset)
- asset_type
Related links (site_web_asset)
- asset_url

Figure 2.4 Mapping of site_web_asset attributes to page elements

strategist can look at developed wireframes and group logically related pages and content to derive unique content types.

Content Elements: Each content type is comprised of elements (sometimes referred to as attributes). Elements bring the actual structure for each content type. Once unique content types are identified, the content strategist can identify the properties of the content type and add them as content elements. For instance, product description content type consists of product title, product image, product brief, and product summary as content elements.

Structure-Enhancing Elements: A content strategist can add other structural elements for these content types. Content elements related to metadata, audit fields (such as author, date fields), category, SEO, and analytics may make the content type easily manageable and discoverable. Sometimes we may also need to add content elements based on business requirements. For instance, Sort by, to filter by attributes, can be added to the content type based on requirements.

Establish Relationships: Now that we have defined all needed content types for the given solution, the next step is to identify and establish relationship among those elements. In our earlier example the product description content type may have too many relationship with product image content type, since a single product description can contain many product images.

Flexibility and Reusability: A content strategist can think of additional content elements that may further enhance the flexibility and reusability of the content type. For instance, we can add a content element "Product text," which can accommodate free-form text. In the future this field can be used to add another product description (such as a mobile version of product description). Look at the reuse potential of a content type and add any additional that which can boosts its reusability.

Support for content models in various CMS Every CMS has its own logical unit of content that forms the content-building blocks. We can leverage those logical building blocks for content modeling. For instance, in WordPress, the logical unit is posts. We can leverage custom post types, custom fields, and custom taxonomies for content model in WordPress. Drupal 8 provides fields and content types for content modeling.

Content Reusability Content reusability is all about crafting techniques to reuse the content across various pages, scenarios, product content, devices, contexts and channels.

Content reusability is one of the key elements of content strategy as it contributes to many of the content strategy goals such as reduced cost, increased productivity, and faster time-to-market through development of modular content.

Content modeling provides much needed structure to reuse the content. We can identify opportunities for content reuse while defining content models. Reusability also depends on the content granularity.

Key advantages of content reusability are:

- It promotes content reuse increasing author productivity.

- It provides easily shareable content for various channels, devices, and site areas.
- It reduces overall content authoring and maintenance cost.
- It reduces overall publishing time bringing content faster to market
- Adopting a reusability strategy creates modular content, making content reusable across various channels.
- It optimizes translation efforts.
- It minimizes content migration efforts.

Main elements of content reusability strategy Before going into finer details of content reusability, let us look at the main constituent elements of the reusability strategy:

- **Content chunk:** Sometimes referred to as content fragment, it is an independent, modular, self-contained, logical piece of content. Chunks form the central piece for content reusability.
- **Content granularity:** This refers to the level or size of the content for a given context. For instance, in product content scenario, a product image forms the lowest granular level for the product content. Content fragments for product brief, product short description, product features, and product long description are coarse-grained content.
- **Reusability governance:** The governance defines processes, best practices, and tools/techniques that can be used for implementing and sustaining reusability initiatives.

Prerequisites for content reusability strategy In order to implement a robust content reusability strategy, we need to have the content ecosystem ready so that it can fully support and sustain the strategy.

- Centralized content repository: Enterprise-wide content should be created and stored in a centralized content repository containing integrated content from all content sources. A centralized content repository also eliminates content silos.
- Availability of content templates: The content tools such as CMS should support templates for authoring and presenting content.
- Availability of metadata hierarchy: Enterprise metadata should be defined and should be available for tagging. Normally we will leverage in-built metadata and tagging support available within CMS or use enterprise-wide taxonomy for this.
- CMS tool: Availability of a CMS tool to implement automation initiatives such as automatic metadata tagging.

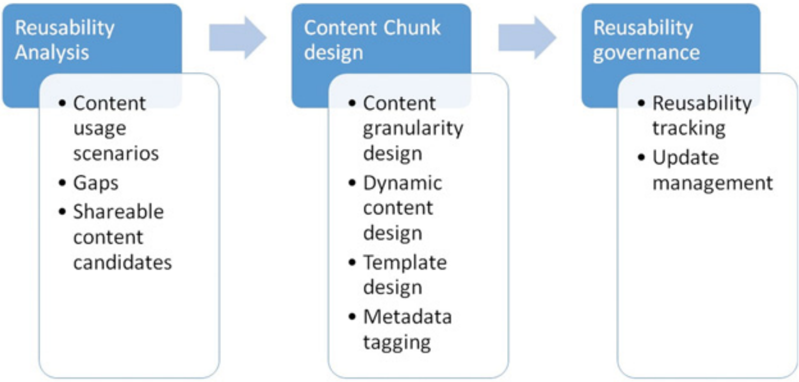


Figure 2.5 Content Reusability Strategy Steps

Implementing content reusability strategy Detailed steps of content reusability strategy are depicted in Figure 2.5. The first step in creating content reusability strategy is to analyze the content requirements and the inputs from content-auditing exercise. As we have seen earlier, content and requirement analysis is part of content strategy planning phase; we can use the insights gathered from that phase. This gives the idea about current content usage scenarios, existing gaps, content-sharing features, and other aspects related to content reusability.

As a next step, we will determine the content granularity needed to create the reusable content fragments. Content fragments can be at section level or at page level. For instance, in the case of a product content, creating content fragments for content short description and content long description at section level makes sense, as these content fragments form logical entities that can be used across site sections and channels; whereas in case of FAQ page, a content fragment could be the entire page section, as there is little opportunity to further decompose the FAQ page content for reusability. Content-authoring templates should support the creation of content fragments.

As most of the enterprise platforms are increasingly becoming dynamic, a static content fragment would not have much reusable potential. So there are other variants of this "fragment/chunking technique" wherein the chunk provides provision for "placeholder" elements, which will be replaced at runtime based on the context. For instance, user name, user role, time zone, currency value, and unit value are ideal candidates for personalizing the user experience, and hence we can use placeholder elements for them.

Once we have identified the proper content chunks and created them using templates, the next steps is to tag them with metadata. The metadata tags should identify the keywords, context, topics, security elements, target device, channel, and audience for the content. Metadata acts as descriptors and marker elements that can be used by presentation engine or search engines to pull the relevant content. Content along with semantic metadata has a multifold impact on the reusability strategy:

- A presentation engine can get the relevant and personalized content based on the user context (user roles, preference, devices). It can apply filters to selectively include the optimized content.
- Search engines can retrieve the relevant content based on its metadata.
- It is possible to improve the author productivity through automatic retrieval of matching content chunks based on metadata relevance. CMS system can detect the closest matching content fragment for reuse based on metadata. Hence a content chunk can be reused across multiple content pages during authoring.
- Dynamic and automatic content aggregation: On the presentation side, entire pages or large page sections can be automatically created through automatic content aggregation based on metadata. For instance, an entire product page can be constructed by retrieving all the relevant content fragments that match the product metadata. This technique not only helps in reusing content fragments across multiple presentation pages but also automates the page creation process at runtime.
- Automatic discovery: Metadata enables automatic relevant content discovery for the structured content.

Chapter 6 discusses the metadata and content tagging.

Content chunking is one of the most effective techniques to achieve content reusability. It involves breaking down the larger content into modular, reusable content chunks. We will discuss content chunking in detail in Chapter 5.

As a part of reusability governance, we use "reusability tracker" to map the content to the pages where it is used. This tracker helps us perform impact analysis and make easier content updates.

Omni-Channel Publishing and Adaptive Content During content design and creation phase, a content strategist has to ensure that created content is optimized for all channels. The basic attributes of Omni-channel-optimized content are:

- Easily discoverable: Content should be tagged with appropriate metadata tags for easy identification and personalization.
- Properly categorized: Content should be categorized into logical categories.
- Adaptive design: This makes content render optimally on all channels. We are going to discuss this shortly.
- Reusable: A chunking strategy should be adopted to create logically independent content fragments, which can be used across content scenarios. We have already discussed this in the previous section.

Adaptive content Adaptive content is the key for achieving Omni-channel publishing.

Adaptive content is essentially a structured and device-independent content, which works seamlessly across various devices and channels through adaptation techniques. It automatically adjusts to various environments and contexts through rendition of best-fit content variants. Adaptive content contains in-built labels/semantic metadata that can be used to filter the content, and it can be used to create the optimal content variant for a given context. We can create content once and publish it anywhere, in any format, using adaptive techniques.

Attributes of adaptive content Let us look at the main attributes of an adaptive content:

- **Channel agnostic**: Content should be stored in a standard format independent of the target channel. Presentation markup, styles, JavaScripts, and navigation logic should not be part of core content. This loose coupling between core content and presentation elements would provide the flexibility to reuse the core content on various devices and in various presentation formats.
- **Semantic metadata**: Content should be tagged with metadata that describes the content. A content consumer would use metadata to get the relevant content. An end-user would also indirectly use content metadata through content search to discover relevant content. We will discuss the content metadata strategy in detail in Chapter 5.
- **Modular content structure**: Content should contain its *rendition information for all delivery channels*. For instance, the content chunk can specify how it would be displayed in mobile device and on a tablet. If there are any embedded digital assets (such as image or video), it should provide alternate text or description or transcript to display on channels where the asset cannot be viewed. Similarly, content variations may help us in choosing the best-fit content for a given context and environment. Modular content may also reduce the translation cost and help in providing targeted and personalized content.

Need for adaptive content Besides making the content Omni-channel enabled, adaptive content offers numerous advantages for a content strategist:

- It makes your content independent of the presentation platform. It offers optimal user experience on various presentation formats such as desktop browser, mobile devices, wearables, and so forth.
- Content adapts based on the context, thus helping the digital marketing effort in delivering personalized and targeted content delivery (thereby achieving other business goals such as increased site traffic, conversion rate, cross-selling, upselling, etc.)
- It helps reuse content for various scenarios.
- It enhances relevant information discovery through tagged metadata.

Creating adaptive content

Prerequisites We have already seen various content analysis activities during the content strategy planning phase. These activities are also required while designing adaptive content for a solution domain. The inputs for creating adaptive content come in three types:

- Context Insights: We need to gain insights about the target audience, user personas, customer needs and expectations, customer's journey, and access devices. We can get this by analyzing use cases, customer mapping, surveys, user interviews, and similar activities.
- Analytics Insights: In addition to this we should also use Web analytics to gain insights into customer behavior, actions, click stream, and so forth. We can use Web analytics software to track user actions throughout user's journey.
- Business Insights: The organization is likely to have specific goals for content and online channels. The goals may include, for example, increasing online sales conversion, reaching out to larger audiences, and so on. We need to understand the specific KPIs and success metrics expected from content.

Once we get good understanding of these three, we can develop the business rules to deliver personalized adaptive content that can help businesses meet the intended goals.

Developing adaptive content

Step 1: Use a chunking strategy. The first step in creating adaptive content is to identify the content usage scenarios and use a chunking strategy to serve content optimally. Based on the content needs for a user persona, we can identify the content types and modular content chunks. We create content by its constituent chunks instead of creating a single monolithic content. Content chunks not only enhance content reusability but also enhance content structure at a granular level. This allows for finer control in using the content chunks in all applicable scenarios.

Step 2: Add tags/labels and semantic metadata for the content chunks. Metadata and relevant tags should be tagged at a content chunk level. Metadata plays a key role in connecting relevant content to an appropriate context. Metadata will be discussed in detail in Chapter 6.

Chunks help us create modular content and provide us with the flexibility to sequence and combine individual chunks in any desired way for a given context.

Step 3: Use Adaptive rules to create content variants. This includes content variations, asset variations, and content reordering, which get triggered based on configured rules. The rules will reflect the context in which the content is rendered, such as:

- Device on which content is rendered
- The location at which content is rendered
- User profile attributes or user segments for which content is rendered
- User's purchase history or browsing history

- Process lifecycle stage during which content is rendered
- Date and time at which content is rendered

These adaption rules can also be used to personalize the user experience. One such example is given in the next section.

Location-based rule for retail content.
Rule 1.0: "If the user is located near one of the retail stores"

Content Presentation: We can provide the label "You can find the following offers at this store" and use the content chunks to list all the sales offers for that nearby store.

Rule 1.1: "If the user accesses the Web site from a remote location"

Content Presentation: We can provide the label "You can find the following offers for this season" and use the content chunks to list all the generic offers. We can add another content chunk that provides a listing of all nearby stores.

Device-based rule for retail content:
Rule 2.0: "If the access device supports video"

Content Presentation: We can provide the label "Check out our video for product installation instructions" and then render the content chunk with video

Rule 2.1: "If the access device does not supports video"

Content Presentation: We can provide the label "Listed below are detailed instructions to install our product" and then render the content chunk that list all the installation instructions. (Instructions should be part of the "video transcript.")

Thus we can offer highly relevant content for a given context using combinations of rules. We can combine the rules related to user's context (location, date, time, channel, and device), user persona (user segments, user preferences, interests, likes, etc.), and insights from user journey (purchase history, click stream analysis, buying stage, touch point etc.).

Responsive web design (RWD) vs. adaptive web design (AWD) Both RWD and AWD aim to create seamless Omni-channel experience for Web sites. There are, however, subtle differences in which they go about doing it, as shown in Table 2.5.

Adaptive content support in various CMS Let us look at the supporting features provided by various CMS to create adaptive content.

WordPress:

- We can create structured content objects through a custom post types feature, custom metadata, custom taxonomy, and custom fields. They can be used to model a content and create structured content chunks.
- We can leverage the shortcode feature to refer to the media markup using its ID. This can be used to dynamically create the media markup at runtime based on presentation requirements.

Drupal:

- We can leverage Field API to model granular and structured content.

Table 2.5 RWD vs. AWD

	Responsive Web Design (RWD)	Adaptive Web Design (AWD)
Key focus area	RWD mainly focuses on presentation components such as fluid layout, grids, and images for Omni-channel enablement	AWD mainly targets the back-end content to achieve Omni channel. It uses adaptive techniques, rules, and progressive enhancements to make the content Omni-channel enabled.
Reusability area	Helps in reusing the same Web site for catering to different devices	Helps in reusing content across channels and devices
Target platform	Mainly mobile devices, desktop browsers, tables, and PDA	In addition to various devices, adaptive content also adapts the content based on the context (user persona, purchase history, etc.)
Dynamic elements	Page layouts and grids change based on target device specifications. As such, content itself does not change.	Content varies based on context and scenarios. Can also work with responsive layouts. In this case, content itself changes.

- A filter system can be used to store the core content and add the presentation markup during content rendition.
- View modules can be used for providing different views of the same content.

Note: The adaptive support for various CMS mentioned above is for educational purposes only. It is possible to use other alternative features and other CMS products to implement adaptive design.

Case study: Adaptive content for product description Let us look at a case study of adaptive content for product page data. As we know, product data contains a lot of information such as product brief description, product short description, product long description, product features, and so on. Entire product information cannot be displayed for all audience and on all devices. In this scenario we are introducing two filters/labels that can help us in providing content variations that can adapt to the device, based on a target audience. We will also see how individual content chunks can be reused to build a larger adaptive content. The content model for product data is shown in Figure 2.6.

In this example we can create six content chunks one each for product briefs, product images, product short description, product long description, product brief features, and product full features. The elements/attributes of these content types are also depicted. As we can see, we have "metadata" as an essential element for each of these content types. Except for product brief, all other content types also have "audience" and "device" elements. "Metadata," "Audience," and "Device" are the labels or filtering attributes that can be used to create content variants based

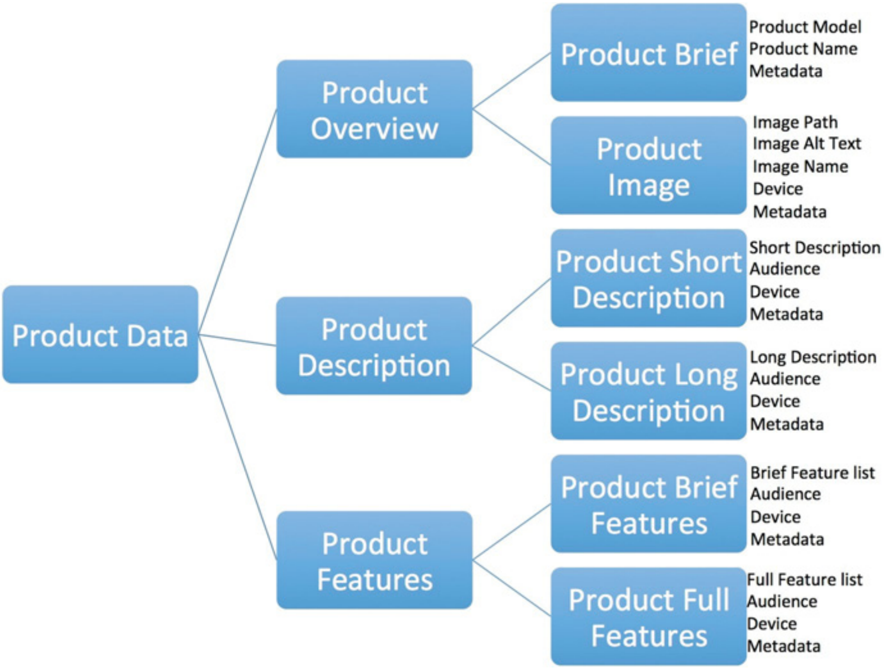


Figure 2.6 Product Content Model

on context. "Metadata" element can also be used to hold other information such as sequence value, product category, author, page section, language, geography, etc.

The assembled content XML is shown in Figure 2.7.

We have seen sample values for audience and device elements in this example.

When a consumer accesses the product data from a mobile device, the resultant content may contain only product briefs, product image, product short description, and product brief features. This may be achieved by using "audience=consumer" and "device=mobile" filters on the assembled XML. Product long description and product full features are shown only when consumer accesses it from a desktop or a tablet. For a "reseller" user, product full features will be filtered on all devices.

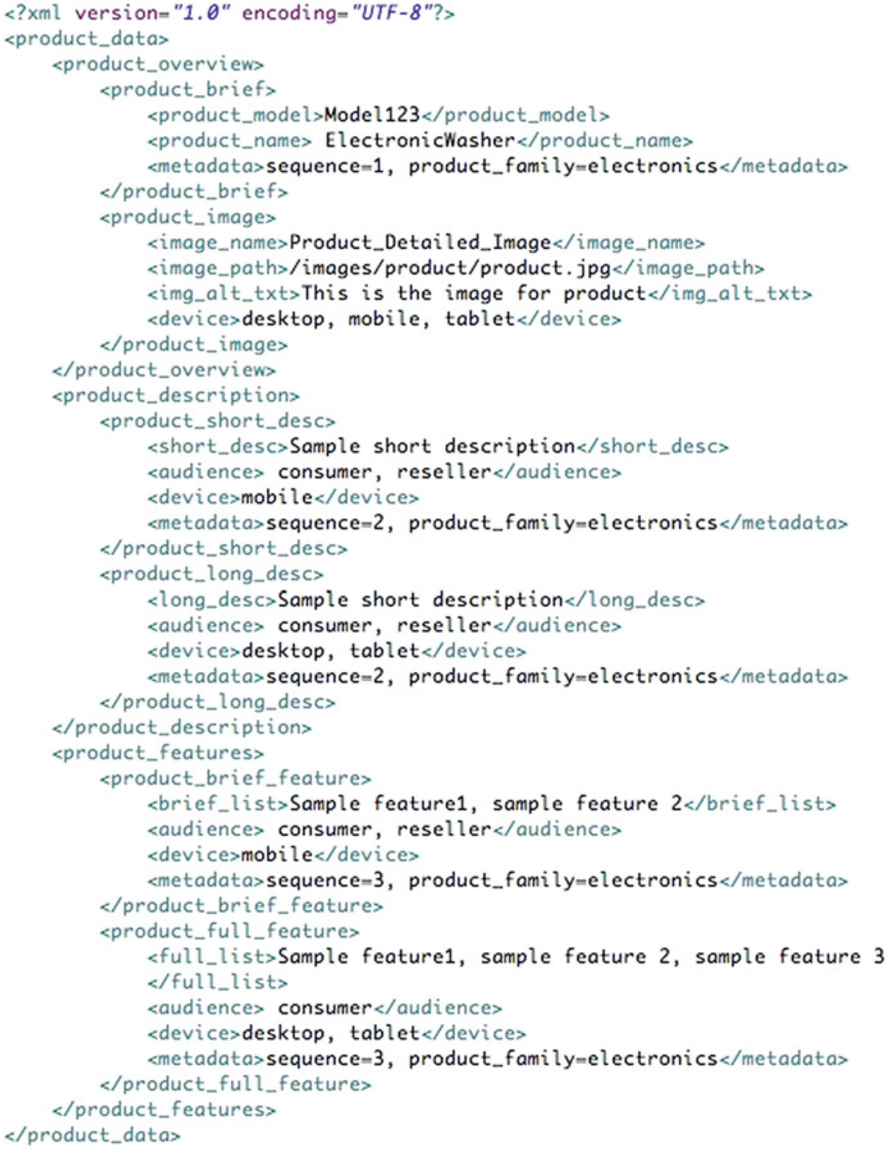
In this example we can see the reusability of content chunks and filtering of content chunks based on labels to create context based variants.

Web editorial calendar This calendar is a key tool that specifies various details of content publishing, timelines, content milestones, campaign schedules, content syndication timelines, and other content timelines.

An editorial calendar aims to improve overall content quality iteratively and align content milestones with business priorities. It provides the general execution plan for the implementation of a content road map.

A sample editorial calendar for products Web site is shown in Table 2.6.

We can see the planned activities on each page for the month of January, February, and March.



```
<?xml version="1.0" encoding="UTF-8"?>
<product_data>
    <product_overview>
        <product_brief>
            <product_model>Model123</product_model>
            <product_name> ElectronicWasher</product_name>
            <metadata>sequence=1, product_family=electronics</metadata>
        </product_brief>
        <product_image>
            <image_name>Product_Detailed_Image</image_name>
            <image_path>/images/product/product.jpg</image_path>
            <img_alt_txt>This is the image for product</img_alt_txt>
            <device>desktop, mobile, tablet</device>
        </product_image>
    </product_overview>
    <product_description>
        <product_short_desc>
            <short_desc>Sample short description</short_desc>
            <audience> consumer, reseller</audience>
            <device>mobile</device>
            <metadata>sequence=2, product_family=electronics</metadata>
        </product_short_desc>
        <product_long_desc>
            <long_desc>Sample short description</long_desc>
            <audience> consumer, reseller</audience>
            <device>desktop, tablet</device>
            <metadata>sequence=2, product_family=electronics</metadata>
        </product_long_desc>
    </product_description>
    <product_features>
        <product_brief_feature>
            <brief_list>Sample feature1, sample feature 2</brief_list>
            <audience> consumer, reseller</audience>
            <device>mobile</device>
            <metadata>sequence=3, product_family=electronics</metadata>
        </product_brief_feature>
        <product_full_feature>
            <full_list>Sample feature1, sample feature 2, sample feature 3
            </full_list>
            <audience> consumer</audience>
            <device>desktop, tablet</device>
            <metadata>sequence=3, product_family=electronics</metadata>
        </product_full_feature>
    </product_features>
</product_data>
```

Figure 2.7 Assembled Product Content XML

Effective content design While creating content, its impact and effectiveness on the end-user should always be considered as one of the main criteria. Figure 2.8 provides four dimensions of effective content.

- **Customer**: We need to understand the expectations, needs, culture sensitivities, user behavior, and navigation friendliness from customer standpoint. Based on

Table 2.6 Editorial Calendar for Products Web Site

	January	February	March
Featured content	Most popular products	Products sales offer	Personalized recommendations
Products Home Page	Products carousel	Products discussion	Products comparison
Products Wiki	Two articles per month		
Products Collaboration Page	Product support forum	Product technology forums	Product communities
Home Page Videos	One video per week	One video per week	One Vvdeo per week

this, we can design the tone, message, style, and information architecture elements of the content. Persona analysis and user journey maps (done as part of content requirements phase) would provide vital insights about the content needs.

- **Market**: We need to do competitive analysis to understand the current market trends and market drivers and choose the best-in-class content strategy to meet the market expectations.

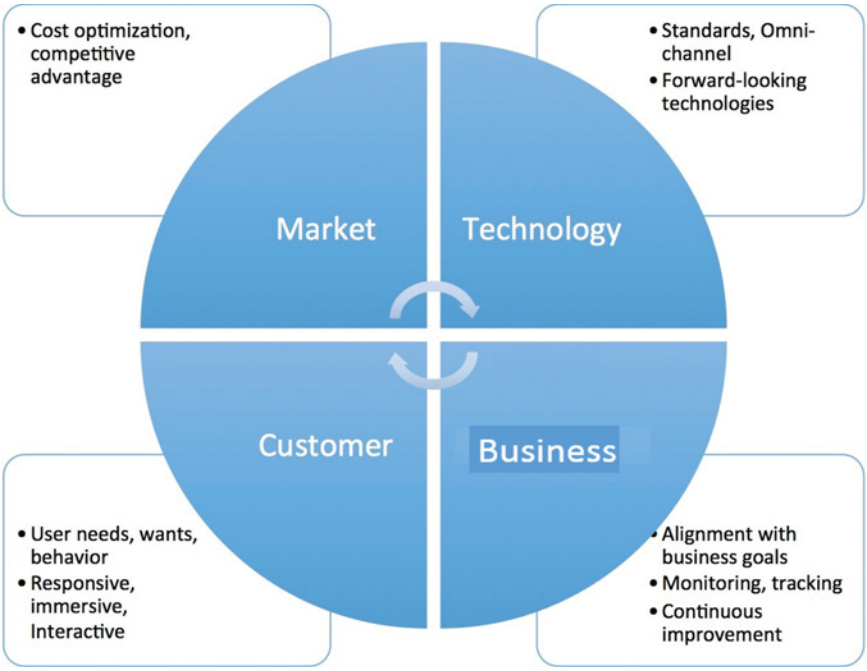

• Cost optimization, competitive advantage
• Standards, Omni-channel
• Forward-looking technologies
Market
Technology
Customer
Business
• User needs, wants, behavior
• Responsive, immersive, Interactive
• Alignment with business goals
• Monitoring, tracking
• Continuous improvement

Figure 2.8 Effective Content Dimensions

- **Technology**: We need to incorporate various technological aspects such as multi-device enablement, adaptive and responsive design, single-page application, and collaborative content to enhance user experience. Use the industry best practices to provide best performing content.
- **Business:** Align content with business goals and objectives. Devise ROI metrics and leverage analytics and monitoring tools to track the content usage and effectiveness.

We can design effective content with these four dimensions. Content should be able to meet user expectation and should be accessible across various delivery channels. Users should be able to access the content in any format (HTML, PDF) on any channel (Web, mobile, tablet).

Customer/user-centric design Based on inputs from user persona and user journey analysis, we need to have the design catering to users' expectations and needs addressing any existing pain points. User-centric design approach is adopted in all phases of solution development. It includes activities such as persona/journey analysis, user research, experience design, implementation, and end-user testing and continuous monitoring. As a part of user research, we may interview end-user groups and have a focused discussions and discovery workshops to understand their expectations, needs, and challenges. Using this information we may model user personas that provide a realistic representation of user groups. We may also benchmark the design and perform competitive analysis to gauge the strengths and weaknesses of the design. The key goal of user-centric design is to match or exceed user expectation and communicate a positive brand perception. During the process we may also address existing users' challenges, simplify the processes, and optimize the task completion time. It is important to adopt a messaging strategy that communicates messages persuasively and consistently with specified branding guidelines.

After initial content deployment, we need to continously monitor the content through Web analytics. Analytics reports provide insights into content usage, asset download rate, ease of content discovery, bounce rate, and other user behavior attributes. It is important to solicit user feedback through surveys, interviews, and other feedback channels.

In the content road map, we need to include user engagement and collaboration features such as blogs, wiki articles, and forums. Social and collaobration features may actively engage end-users and help us harness collective intelligence.

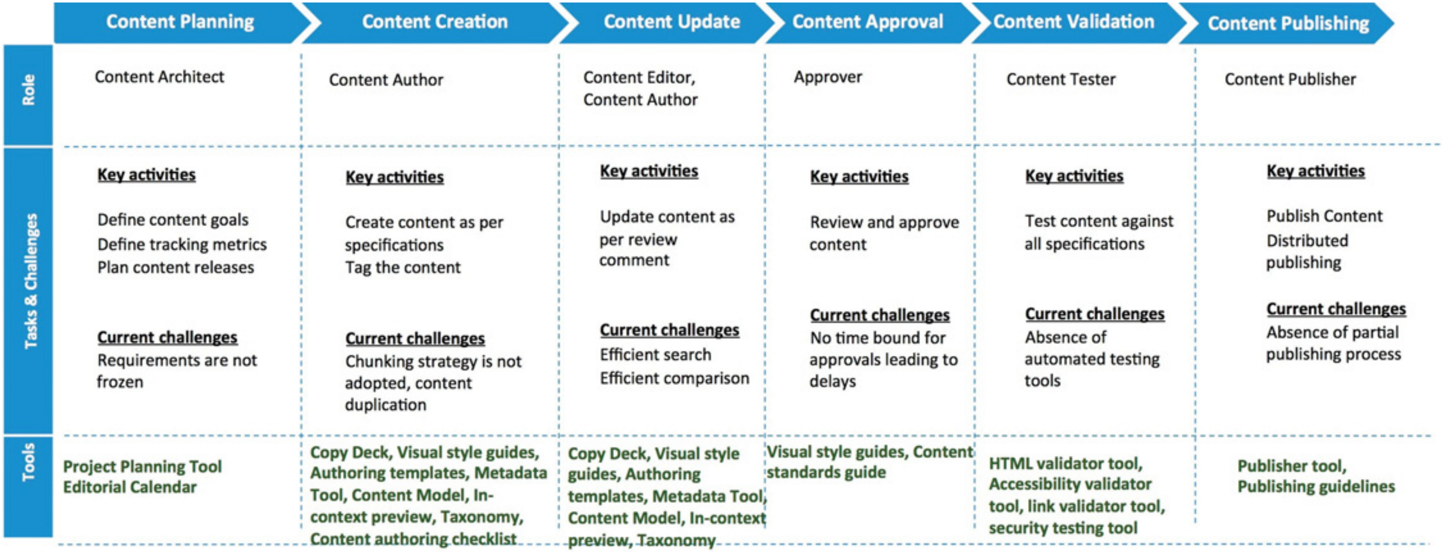
Content Governance Content governance is all about defining roles, responsibilities, and processes to efficiently manage content through its entire lifecycle. Content governance defines the role-responsibility matrix and processes and standard operating procedures (SOP) to handle content scenarios.

Governance is key for the long-term success of content systems. It brings much-needed structure and discipline for handling changes and provides a response framework to handle exceptions. The main drivers for content governance are:

- **Uniform and consistent branding:** Content governance defines the Web site branding guidelines, UI/UX standards, and navigation structure that help in creating a consistent user experience.
- **Defining standards:** The content governance body in close collaboration with content strategists and architects will define the content standards for the organization. Standards for content storage, content exchange, content integration, content services, and other content touch points that are part of user's content journey will be defined and documented.
- **Well-defined processes for handling content operations:** Content governance defines the role-responsibility matrix – who does what. This creates a standard operating procedure (SOP) for handling content processes in the most optimal way.
- **Increased productivity:** This can be achieved through well-defined and agreed-on task ownership.
- **Enhanced content quality:** Content governance defines content verification and validation processes. Through proper content verification and review processes entities can deliver high-quality content to meet business goals.
- **Enhanced user experience and active engagement:** Content governance defines various continuous improvement measures. Incorporating user feedback is one of the main continuous improvement measures. This, along with other improvement measures such as collaboration processes, may result in improved user experience and enhanced user engagement.
- **Satisfying business SLAs:** Governance plays a key role in meeting business SLAs related to performance, scalability, and continuous availability. Various processes related to business continuity, disaster recovery, and monitoring may be defined as part of the governance process.

Defining content governance The first step in defining the content governance process is to understand various content roles and map content activities with corresponding roles. A content strategist can understand the existing processes, challenges, roles, and content activities during content requirements exercise. Content activities may be performed by various roles such as content super admin, content author, content reviewer, workflow admin, site admin, and so forth. It is important to understand all security requirements and associated roles. If the roles are not defined at the outset, they must be defined while content activities are under way. A sample mapping of content roles to content activities for a content-publishing process is depicted in Figure 2.9.

At the same time, it is important to compile the list of all content activities and content functions that need to be secured. The content analysis phase (during content strategy planning phase) provides good inputs for this activity. Some of the core secure content activities are creation of content templates, creation/update of content, publishing content, configuring content workflows, scheduling publishing jobs, content versioning, content purging, defining site taxonomy, and so on.



	Content Planning	Content Creation	Content Update	Content Approval	Content Validation	Content Publishing
Role	Content Architect	Content Author	Content Editor, Content Author	Approver	Content Tester	Content Publisher
Tasks & Challenges	**Key activities** Define content goals Define tracking metrics Plan content releases **Current challenges** Requirements are not frozen	**Key activities** Create content as per specifications Tag the content **Current challenges** Chunking strategy is not adopted, content duplication	**Key activities** Update content as per review comment **Current challenges** Efficient search Efficient comparison	**Key activities** Review and approve content **Current challenges** No time bound for approvals leading to delays	**Key activities** Test content against all specifications **Current challenges** Absence of automated testing tools	**Key activities** Publish Content Distributed publishing **Current challenges** Absence of partial publishing process
Tools	Project Planning Tool Editorial Calendar	Copy Deck, Visual style guides, Authoring templates, Metadata Tool, Content Model, In-context preview, Taxonomy, Content authoring checklist	Copy Deck, Visual style guides, Authoring templates, Metadata Tool, Content Model, In-context preview, Taxonomy	Visual style guides, Content standards guide	HTML validator tool, Accessibility validator tool, link validator tool, security testing tool	Publisher tool, Publishing guidelines

Figure 2.9 Mapping Content Roles to Activities, Challenges, and Tools

Table 2.7 Content Role Responsibility Matrix

	Create/edit content	Approve content	Publish content	Content administration
Content Author	Yes	No	No	No
Content Reviewer	No	Yes	No	No
Content Super Admin	Yes	Yes	Yes	Yes
Content Publisher	No	No	Yes	No

Once we have listed all the content roles and compiled a list of content activities, we need to create a role-responsibility matrix (which will be later translated into an access control list). A sample role-responsibility matrix is shown in Table 2.7.

The role-responsibility matrix has to be implemented in the underlying CMS through built-in permission model and role-based access features.

Additionally, content governance is also responsible for defining the process for these content elements:

- Defining the framework for content standards used in the solution.
- Taxonomy management process: The process provides details on adding, updating, and deleting terms (related to content) from taxonomy. It may also provide details about the usage of taxonomy terms for tagging and categorizing content and assets.
- Rights management process: The process specifies how the access rights and permissions are managed.

2.4 CONTENT STRATEGY CASE STUDY

So far, we have seen various concepts, phases, and examples of content strategy from different dimensions. Let us put those concepts to action through this case study example.

Fashion E-Tailor Case Study

Background Fashopping is a popular online e-commerce Web site specialized in selling high-end fashion apparels and accessories. The site was developed five years ago and its content was rendered mainly on desktop browsers. The organization wants to expand to other geographies and reach out to more consumers.

Fashopping is facing challenges on multiple fronts: competitors are rolling out better content market quickly; consumers are facing difficulties in accessing content from mobile devices; content processes are complex, causing delays. As a result of this, the organization is steadily losing ground to its tech-savvy competitors.

The CEO has hired a content strategist to redesign the content and bring Fashopping back on a competitive track.

A content strategist may adopt the four-phase content strategy exercise we discussed earlier. Let us look at the activities in each of those phases.

Table 2.8 Fashopping content analysis

Target audience	Young consumers in the 17–26 age group Tech-savvy audience who need Omni-channel access and inspiring content
Main user needs and expectations	Provide latest fashion brands early to the market Provide Engaging, personalized, and immersive user experience Allow Access to the online channel anytime, anywhere, from any device Spawn the conversations and discussions to internal and external social media channels
Competitive analysis	Most of the competitors had embarked on a digital transformation and have created forward-looking Omni-channel content Competitors have adopted responsive web design (RWD) and adaptive design techniques for Omni-channel enablement
Content inventory	The site mainly consists of Web content for 45 site pages authored in English. The pages use 590 product images. There are mainly two workflows for content authoring and publishing.

Planning the content strategy for Fashopping In this phase, the content strategist will analyze existing content and interview business stakeholders to understand the goals and identify gaps.

Fashopping content analysis: Given the company's fashion-focused content, the site aesthetics, voice and tone, and messaging play a primary role in the success of the site. Most of the site visitors are young adults who expect an engaging, interactive, and immersive experience. Based on the user research and surveys, the content strategist has analyzed the following factors (Table 2.8).

Fashopping Business Goals: The content strategist interviews business stakeholders and identifies the primary business goals.

- Position Fashopping as a primary shopping destination for fashion brands.
- Enhance user experience to motivate young audiences and influence their purchasing decisions.
- Convert first-time visitors to brand ambassadors through personalized and engaging experiences
- Increase site visitors and drive their journey to end on a checkout page for increased conversion
- Make content available anytime, anywhere, on any device
- Make content more usable and easily discoverable.

User Personas: The main user personas for Fashopping are fashion shopper, visitor and support executive. Fashion shopper persona mainly consist of young adults in the age group of 17-26 and are tech savvy audience who accesses the site mainly from mobile devices. They are interested in shopping in trending fashion products and are heavily influenced by social recommendations. They want quicker checkout process with least number of steps in shopping process. Visitors mainly access the public web pages and heavily use search. The home page, landing page and search

Table 2.9 Fashopping Content Mapping

	Challenges	User goals	Business goals	Content Type	Content Gaps
Product Landing page	Too much of cluttered information, Lack of immersive experience	Check out latest fashion apparels	Promote premium brands	Product content, product images	Absence of adaptive content, Absence of categorized content (such as trending fashion etc.),
Shopping Page	Complex shopping process	Purchase the selected fashion apparel	Ease the product purchase experience	Product recommendation content,	Absence of adaptive content

function should be optimized for visitors. Support executives mainly collaborate with customers over chat and over phone.

Content Mapping: The content mapping exercise revealed the gaps in existing content. The Fashion shopper persona's shopping journey was modeled to map the content and identify the gaps. The content mapping for fashion shopper's shopping journey is depicted in table 2.9.

Content mapping exercise revealed gaps in Omni-channel content and categorized content. These gaps and challenges were taken into consideration along with other identified gaps.

Gaps in current state content ecosystem

During content mapping exercise and content analysis, content strategist identified following gaps in existing content eco-system:

Technology gaps

- Content is mainly written for desktop browsers; we need to make it work for all devices.
- Absence of metadata and tagging leading to challenges in content discovery
- Absence of SEO and web analytics leading to poor site visibility and user behavior tracking
- Absence of DAM system for digital assets
- Improve the information architecture and site taxonomy
- Absence of templates causing minimal content reusability, duplicate effort and increased time-to-market
- Legacy CMS which does not support contemporary content standards and technologies. Digital assets are managed within CMS
- Lack of focus on social and collaboration channels.

Challenges in end-user experience

- Users are facing challenges in finding right content with in-consistent navigation and user experience.

- Site content does not work optimally on users' mobile devices and hence users are switching to competitive sites who offer mobile friendly site or mobile apps
- Inconsistent user experience due to absence of visual style guide and branding standards.

Defining the content strategy for Fashopping

Based on the analysis of the business goals, content strategist would distill the strategy in *core strategy* statement as follows:

"*Fashopping content should empower its users to do more with the site through responsive, forward-looking, engaging and easily findable, single-source fashion content on all channels*"

Content strategist would also define other themes and focus areas for the overall content improvement:

- Streamline content processes to improve content freshness and for faster content updates
- Evaluate CMS products and chose the best-of-breed CMS to suit the requirements and which supports metadata. Use the new CMS to create re-usable templates, reusable content and create Omni-channel adaptive content. Leverage separate DAM for storing and tagging assets. New CMS and DAM systems should be able to support localization and multi-site management features
- Re-design the user experience through visual style guides, consistent brand elements
- Improve information architecture for easier content discovery and faster site navigation
- Use SEO, analytics for driving user traffic to the site and track user actions
- Enable social and collaboration features such as blogs, forums for active user engagement
- Establish content governance to define role-responsibility matrix and define structured processes to handle regular content operations

Content strategist would also layout the roadmap for the content re-design exercise. The key elements of roadmap is given below:

Business Goals	Quarter 1	Quarter 2	Quarter 3
User experience enhancement	Define site wide visual guide Simplify content publishing process Use templates for consistent page structure and layout	Implement adaptive and responsive design Integrate with external social channels Provide engaging and interactive content to check out and compare trending fashion content.	Continuously improve the content performance Setup forums and communities to actively engage users

Business Goals	Quarter 1	Quarter 2	Quarter 3
Increase User Conversion	Provide persuasive and influencing content	Use web analytics to track user activities and fine tune content	Setup monitoring and notification infrastructure
Single stop shop for Fashion brands	Consolidate content from all data sources Use new CMS and DAM systems to implement business goals.	Provide improved user-friendly information architecture	Provide personalized and context aware content through adaptive techniques
Easy discover-ability of content	Remove all duplicate and outdated content Tag relevant metadata with content chunks Use social media marketing campaigns to spread the awareness about new site.	Enable faster content search and site search Use SEO best practices to provide more visibility	Continuously update SEO and launch targeted campaigns to enhance content effectiveness.

Content strategy execution for Fashopping

Content strategist oversees the content design and creation activities. Let us look at the key design decisions and how it aligns with the defined content strategy and addresses the identified gaps

Visual re-imagine with optimal content messaging and tone: Entire site content would be re-designed using newly created templates. Style guides, brand elements and visual guidelines would be used to provide consistent user experience and to effectively communicate the brand identity. Branding and visual guidelines specify the brand logos, font format, heading size, image specifications and such. Using contemporary look and feel using crisp and inspiring messages and immersive experience using adaptive images was achieved.

Content Page redesign: Some of the existing pages were very generic making them unusable. Pages were improved to provide the most relevant information with effective usage of multi-media. Instead of a generic contact us page, personalize the page based on user's purchase history (by providing specific contacts for previously purchased products). Include most relevant questions in FAQ page (such as "How do I compare the products?", "What is the return policy?")

Engaging Content: Provide engaging content for users to help them make quicker decisions and convert. In the shopping pages provide addition features such as rating, “compare this to another product”. In the support pages provide easily readable “How-to” content and product demonstration videos. Provide a blogging platform for users to blog about the brand and products.

Content reusability through adaptive content chunks: Content models would be created for the new content structure and re-usable content chunks would be designed. Content chunks use adaptive techniques to cater to all supported browsers and devices.

Metadata tagging: The in-built metadata support in the new CMS was used to define the required metadata for the site. All content chunks were tagged with semantic metadata.

Content creation: Reusable Content templates were designed with the new CMS. Templates would be used to create the modular content chunks. Flexible content models were developed to cater to all content structures of the site. Tagging content chunks with semantic metadata enhanced content discoverability. Adaptive content was designed to support all devices

Content workflows: Workflows were re-designed for optimal content authoring and publishing experience. Redundant and duplicate steps were eliminated and few approvals were automated and few more were made time bound. Localization workflow was introduced for content translation.

Content testing: The content on business-critical pages (such as check out page, home page) were tested through multi-variate testing techniques and A/B testing to ensure that right content and click-to-actions were used for enhanced user impact.

Content collaboration: Using content components and widgets, fashion-related discussions (such as trending fashions, brand discussions) from external forums. The new site would also provide more collaboration channels such as blogs and wikis for enhanced end-user participation.

Content information architecture: Highly structured site maps, easy-to-read content and intuitive navigation structure would be adopted to enhance content discovery

Multi-media content: In order to attract more visitors and enhance site stickiness, multi-media content such as promotion videos, campaign content were featured on home page. Rich media assets would be optimally managed (stored, tagged, retrieved) from the new DAM.

Content SEO and analytics: Web analytics would be used to track the usage of content and other site metrics such as visitor rate, conversion rate, and time on site and such. Analytics reports would be used to fine-tune the content. SEO best practices would be adopted to improve the site and content visibility

Marketing campaigns: A series of focused campaigns would be launched for each of the user segments. Campaign content provided the differentiating factors of fashopping and contained latest fashion trends and popular categories. Promotion campaigns on external social media sites and search engine marketing (SEM) was also used to create awareness and drive traffic to the new site. Carry out creative marketing campaigns to keep the visitors interested. (Such as crowd-sourced fashion design contests, voting campaigns etc.). Perform targeted email marketing based on purchase history.

Table 2.10 Fashopping Editorial Calendar

Content Task	Owner	Publish date	Priority	URL
Create campaign content for home page	Writer 1	10-Nov-2015	1	Fashopping.com/home
Re-design the check out page	Writer 2	25-Nov-2015	1	Fashopping.com/checkout

Omni-channel presence: Adaptive content ensured that content was optimized for all supported devices. A responsive front-end was used along with adaptive content.

Editorial calendar: A well-defined editorial calendar prioritized the tasks and specified the ownership and timelines. An extract of the editorial calendar given in Table 2.10.

Content strategy maintenance for Fashopping

Using the web analytics following content metrics were continuously monitored:

- Page visits: More visits indicate the success of marketing campaigns and other promotion campaigns
- Bounce rate: Lesser bounce rate coupled with higher "Time on site" is a testimony of usable and effective content
- Conversion rate: Higher conversion indicates the effectiveness of check out page and the shopping pages. This metrics needs to be closely looked at to ensure that UI elements and content on the business critical pages are optimized
- Time on site: A higher time on site indicates that content has captured user's imagination and is able to successfully engage with user.
- Social share: End user sharing a site page or content page would boost the visibility and branding of the site.

As a part of maintenance, weekly content updates related to featured content, promotion content would be rolled out to the home page.

2.5 CHAPTER SUMMARY

- This chapter provides details of various concepts and elements of content strategy
- Content strategy aims to make the content more effective and more usable for the target audience
- Business audience, marketing team, UX designers, content architects are the key stakeholders of content strategy
- The key goals of content strategy are: defining content tenets, increasing content efficiency, delivering Omni-channel content, designing efficient

content, defining content governance and optimizing content eco system elements

- Common challenges with content are: duplicate content, content silos, Inconsistent and non-uniform content, Complex and inefficient content processes, Less effective content leading to decreased usability and conversion rate, Content discovery challenges, Slower turn-around times, Content integration challenges, Omni-channel delivery, Challenges with content collaboration
- Main characteristics of good content are content usability, customer centricity, optimal messaging, Conciseness, Findability and searchability, Contextual and relevance, Consistency, Accessibility and Omni-channel enablement, Information architecture and information discovery, SEO friendliness, Content readability, Content messaging and communication, User engagement and user adoption, User information content, Content tracking and optimization and Content presentation
- Content best practices are Content chunk strategy, Deliver personalized and contextualized content, Create user-centered content, Optimize content ecosystem, Adopt content reusability strategy, Identify and eliminate duplicate, unnecessary and outdated content, Intuitive Information architecture and information discovery, Content tracking and optimization and Content presentation
- During content requirement elaboration we will gather requirements for content voice, user personas, digital assets, localization. During this stage we will also gather requirements for Omni-channel content, content analytics, social and collaboration content, non-functional requirements and business related requirements
- Four phases of content strategy are Content Strategy planning, Content Strategy definition, Content Strategy execution and Content Strategy maintenance
- In Content Strategy planning phase, we will perform content analysis, content auditing, content inventory preparation, gap identification, Competitive analysis and benchmarking and Current challenge analysis
- In content strategy definition phase, content strategist will define core strategy, content governance and content roadmap
- In Content strategy execution phase, content strategist will oversee content design and content creation activities
- Content strategy maintenance phase includes regular content updates, patches, monitoring and governance activities
- Key elements of content strategy are content models, content chunks, adaptive content, editorial calendar and user-centric content design
- Content model provides structure for the content and depicts the messaging value of the content. It defines various content types along with their relationship
- For creating content models, we need to find identify unique content types, its content elements and then establish relationship between content types

- Main elements of content reusability strategy are content chunks, content granularity and reusability governance
- Content chunk is a self-contained, independent logical entity, which can be used for creating a topic in structured way.
- Main design considerations for content chunks are Right Granularity, Chunk candidate identification, Metadata and tagging, Content Model, Chunk templates, Chunk XSD and DTD and chunk security
- Adaptive content is essentially a structured and device-independent content, which works seamlessly across various devices and channels through adaption techniques.
- Key attributes of adaptive content are Channel agnostic, semantic metadata, and modular content structure
- In order to create adaptive content we need to first create modular structured chunks, we then need to add labels and metadata and use adaptive rules to get content variations
- Web editorial calendar provides various details of content publishing, timelines, content milestones, campaign schedules and deadlines
- User-centric content design needs to focus on 4 dimensions: customer, market, technology and business
- In the end of the chapter we discuss a case study for fashion e-tailor

Chapter 3

Basics of Content Management Systems

We have already seen the crucial role played by content management in digital platform in the previous chapter. This chapter takes an in-depth look at various aspects related to content and how it fits the bigger digital picture. Understanding content management from various perspectives helps in designing a robust content strategy well suited to realize the desired business benefits, and in developing a more effective digital platform.

We begin with an introduction to content management. We will define content management system, its business drivers, utilities, and challenges in a typical enterprise scenario. We then will look at common roles in a content management scenario and various guiding principles for designing a robust CMS. Other things to be explored are core capabilities, CMS component functionality, and desired attributes of CMS. We then will look at a comprehensive functional architecture of a typical enterprise CMS. The chapter will conclude with a brief look at open-source CMS systems such as Drupal, Joomla, WordPress, and Apache JackRabbit, a JCR implementation.

Enterprise architects, content architects, and CMS developers will find this chapter useful.

3.1 WHAT IS A CONTENT MANAGEMENT SYSTEM?

A content management system (CMS) is a set of technologies that help in creating, publishing, customizing, distributing and managing content throughout its lifecycle.

Content management involves collection, conversion, storage, retrieval, management, maintenance, organization, modification, enhancement, publication, archival and deletion of the content.

Every organization that depends on content (such as e-commerce, digital marketing, etc.) needs content management. Based on the specific needs the content processes, workflows and business rules may vary.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

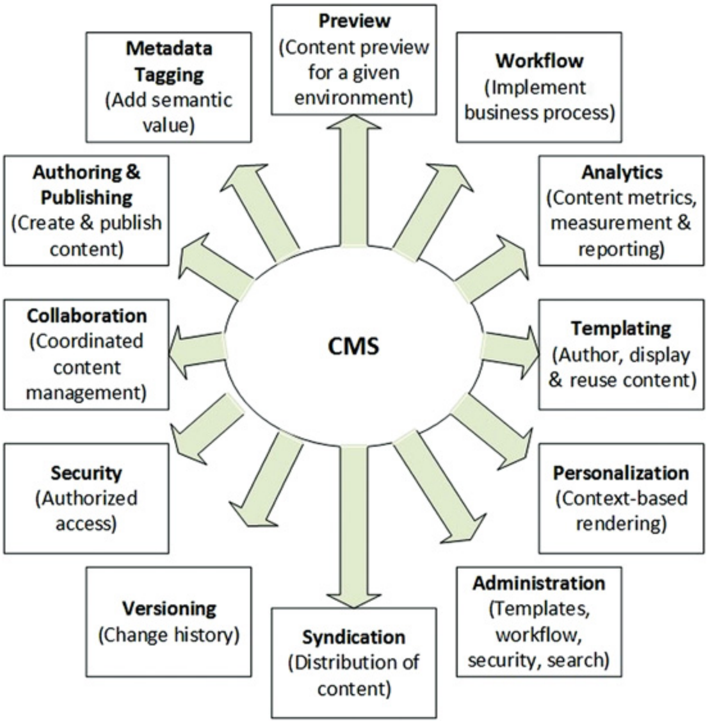

Preview
(Content preview for a given environment)
Workflow
(Implement business process)
Metadata Tagging
(Add semantic value)
Authoring & Publishing
(Create & publish content)
Analytics
(Content metrics, measurement & reporting)
Collaboration
(Coordinated content management)
CMS
Templating
(Author, display & reuse content)
Security
(Authorized access)
Personalization
(Context-based rendering)
Versioning
(Change history)
Syndication
(Distribution of content)
Administration
(Templates, workflow, security, search)

Figure 3.1 CMS Core Features

An example of a basic content management process is a content publishing process: authors for the Web site create digital content; authors and reviewers can update content; designated approvers will review and approve content; once all approvals are done, content is published.

The key features of CMS are depicted in Figure 3.1.

A CMS provides a comprehensive ecosystem to manage content. It provides authoring and publishing templates, metadata tagging, and content preview features for content authoring and publishing. Content workflows can be used to implement the content processes and associated business rules. CMS also provides administration interfaces to manage the templates, workflows, and other functionality. Some CMS also provide in-built collaboration and syndication features. CMS provide role-based access to its functions, and we may also track content metrics through built-in features.

Note: While discussing CMS topics in this book, we mainly focus on CMS focus areas we specified in the first chapter: Web content management, content workflows, Web publishing, asset management, document management, taxonomy, metadata, content security, content analytics, responsive content design, and content performance.

Business Drivers for CMS

There are several motivations for enterprises to choose a CMS. CMS play a crucial role in digital transformation journey for enterprises. The key business drivers for CMS are:

- Efficient content management: This is a natural motivation for using a CMS. CMS provides all required components to manage content for its entire lifecycle.
- Agile and efficient business processes: An enterprise has to respond quickly to the changing customer demands and react to the market dynamics. This needs efficient processes to create, review, and publish content. All redundant and unnecessary steps need to be removed. CMS provides efficient content workflows to model the business processes. Organizational Web sites can quickly adapt to publish new products, services, or corporate strategies. Quicker publishing and content turnaround can help a business take a leading position in the market.
- Management of information explosion: Rapid pace of content creation requires efficient techniques to store, retrieve, and publish content and related assets. We can leverage CMS to manage end-to-end content lifecycle along with is dependent components.
- Content consolidation: CMS can be used to consolidate, extract, and synchronize content from multiple content sources. Consolidated content can then be categorized and tagged so that it is easy to identify relevant content. CMS can render content in a standard format. Integrated content may also act as a single source of truth for content needs.
- Optimal user experience: CMS can be used to create consistent brand experiences with appropriate message, voice, and tone. The CMS enhances the quality of the site by providing an integrated, enhanced, and easy-to-use environment for Web site visitors.
- Business functions realization: CMS helps in realizing various business functions such as digital marketing, campaign management, product information management, etc.
- Faster business responsiveness: The CMS supports the organization in integrating new products and services, in providing quick information updates and through efficient content processes.
- Streamlining information updates: CMS helps in digitizing and streamlining manual processes to have fresh and streamlined content across site.
- Content security: CMS provides better control of content resources and processes. We can implement role-based access and permission model in securing content resources.
- Information quality: CMS can be used to facilitate content quality improvement through content reuse, content security, content metadata, and content structure.

- Return on investment (ROI): Overall ROI can be enhanced through efficient content processes, content reusability, and agile content publishing and reduced duplicate content. The need for higher ROI places CMS on the strategic road map of the organization.

Utilities of CMS

Enterprises can utilize CMS to achieve many technical, business, and operational goals. The following list holds some of the key benefits of adopting a CMS:

- Content and template reusability: CMS facilitates content reusability through creation and publishing of modular and structured content chunks. Content authoring and publishing templates can also be reused to create various site sections.
- Separation of content and presentation: CMS separates core content from its presentation. Core content can be stored in standard format and we can later add presentation (using presentation templates) for specific presentation devices. This separation offers greater flexibility and reusability.
- Efficient publishing: CMS provides efficient and easier content publishing across various sites in different languages. CMS can provide streamlined publishing process and replace manual/inefficient processes. Administrators can use friendlier user interface to model the publishing workflow with minimal dependency on technical teams.
- Consistent information: A CMS ecosystem provides consistent and findable content through usage of metadata, classification, taxonomy, and intuitive information architecture. The content can be used across enterprise.
- Reduced duplication of information: CMS stores all the information into a single repository reducing the risk of duplication of information across business units. This helps in decreasing the maintenance costs and error rates.
- Reduced costs: CMS can help reduce the costs on multiple fronts: reduced authoring cost due to content reusability, reduced publishing cost due to efficient publishing, and reduced site maintenance due to self-service and business-friendly processes.
- Increased site traffic: We can build engaging and interactive user experience using CMS. This would increase the site traffic and can actively engage site users.
- Central repository: CMS forms the centralized content repository. CMS can consolidate content from various enterprise sources and can act as single point of access for all enterprise systems.
- Business-friendly interface user access: CMS offers easy-to-use interface that a business can use for performing various business-critical activities such as publishing, site creation, workflow administration, site administration, access

control, and others. This minimizes dependency on the technical team and acts as a self-service platform that increases agility.

- Version control: We can implement version control for content, assets, and releases. This helps perform version comparison, rollback, and creating release tags.

Let us now look at some of the prominent challenges we would normally face while implementing the CMS for content projects.

Challenges in Implementation of CMS

The common challenges we face while establishing a robust content management system are:

- Content structure variations: As enterprises have content in various structures/formats, we need to design the authoring templates, design templates, and site structure to accommodate all types of enterprise content.
- Existence of content silos: Distributing content among various systems leads to content silos. Duplicate and inconsistent content may exist among various departments, channels, and business groups. Content silos will, in turn, lead to duplicate content and inefficient processes.
- Content migration challenges: In some cases we may need to migrate existing structured and nonstructured content to a CMS system, which requires content transformation, metadata mapping, content category mapping, and automation. These aspects need to be carefully designed to ensure optimal migration of large-scale content. Chapter 9 provides a detailed content migration strategy.
- Challenges in modeling taxonomy and metadata hierarchy: The taxonomy term design and metadata design and hierarchy should meaningfully describe the content and reflect the underlying business requirements (such as product keyword hierarchy, geography hierarchy, user role hierarchy, user group/segment description, etc.). The design should also be flexible to accommodate any future changes.
- Challenges in defining site taxonomy: The site taxonomy should be designed to accommodate the future requirements such as geospecific or language-specific variations.
- Modeling business processes: The business processes related to content such as content review process and content publishing process vary from organization to organization. The content workflows should appropriately model the business processes and implement the underlying business rules.
- Publishing delays: In some scenarios due to the complex deployment structure (such as remote site deployment or deployment of multiple copies of localized content) and complex publishing workflows coupled with complex business rules, there could be a delay in publishing the content to production sites. This

Table 3.1 Poor vs. Optimal content management

Poorly Managed Content	Optimal Content Management
• Processes laden with manual and redundant tasks • Challenges in discovering relevant information • Limited access to latest information • Large silos of content hampering information sharing • Unstructured information and large volume of content delaying decision making • Duplicate and outdated content • Inefficient systems to manage the content	• Lowered paper handling and reduced error-prone manual processes resulting in lowered operational costs • Unified access to derived, personalized, and categorized information enhances decision-making process • Integrated business processes and linking inter- and intra-organizational boundaries leading to increased efficiency • Improved control over documents and document-oriented processes improves performance • Meeting regulatory compliance for records keeping and handling reduces organizational risks • Distributed scalable solution architecture capable of handling anticipated growth

could potentially lead to concerns among business stakeholders and impact release timelines.

- Omni-channel content delivery: As site content is accessed from multiple devices, content developed for a particular delivery channel may not work for all devices and form factors. Hence, content should be designed using adaptive techniques so that it will render optimally in all channels.
- Content performance: The performance of the content can vary based on geography, device, site, and other variations. It is important to test and optimize the content performance for all variations.

The content challenges listed above, if unaddressed, lead to poor content management. The impact of poorly managed content is depicted in Table 3.1:

In the next section we discuss the CMS trends that are gaining traction in modern content implementations.

CMS Trends

Understanding CMS trends helps in designing efficient CMS solutions and provides provisions for future requirements. We have noticed these emerging trends in CMS engagements:

- **Convergence of CMS with portals and search technologies:** We are seeing increasing convergence across digital technologies such as CMS, portals, and search. Most CMS products offer lightweight presentation portals that are seamlessly integrated with native content platforms. CMS portals also provide page layout customization, customer segmentation, personalization, security, and basic information aggregation. We could avoid using a full-fledged horizontal portal product or a separate presentation engine and use the native CMS portal. We discuss a CMS portal case study in content case studies chapter in book support website.

- **CMS as experience platforms:** CMS systems are widely being used as Digital Experience Platforms (DXPs), especially in digital marketing scenarios that provide 360-degree view of customer activities. CMS-based DXPs offer content personalization, localization, Omni-channel content, responsive user experience, collaboration, and similar features to actively engage users. A detailed CMS-based DXP is discussed in Chapter 4.
- **Leaner CMS models:** Lightweight, Web-based CMS models are increasingly used by customers to optimize their time to market. Development time, testing time, and deployment time will be greatly reduced by leaner CMS models. Leaner CMS models ideally fit agile delivery methodology as they help in iterative and faster releases.
- **Flexible business controls and marketing/sales enablement:** Intuitive user interfaces offer greater control to business stakeholders and enable them to configure and perform business-critical activities. Content administration, promotion management, campaign management, workflow management, and search management are some of the features business administrators could use with minimal dependency on IT and technical teams. Marketing and sales functions use CMS for efficiently managing campaigns and promotions.
- **Omni-channel content:** Responsive web design (RWD) is widely used to implement Omni-channel content and mobile-first strategy through flexible layouts and media queries. Adaptive content design is used at content layer to cater to multiple devices. Many CMS platforms may also provide mobile content authoring, device rules, responsive themes, device-specific publishing and other features for mobile enablement. We have already discussed adaptive design in Chapter 2.
- **Social and collaboration:** CMS platforms are offering built-in collaboration features such as blogs, wikis, message boards, communities, and forums to actively engage end-users. CMS is one of the favorite choices for building collaborative knowledge base platforms. We discuss one such case study in content case studies chapter in book support website.
- **Seamless experience across channels:** CMS platforms and its ecosystem components (such as business process management components, user experience components, collaboration platforms and such) can be used to provide an integrated experience across all user touch points. An user can initiate the journey through a desktop device and continue it on mobile channels without the need to re-start the process or re-enter the data. Enterprises can engage the

customers more actively through this seamless conversation across social and mobile channels.

- **Built-in asset and document management capabilities:** Many CMS platforms provide built-in basic document and asset management capabilities that seamlessly integrate with CMS portal, workflows, and content management processes.
- **Analytics-driven personalization:** Insights are gathered about user behavior through Web analytics–based tracking. These insights are employed by segments users to personalize their content and experience.
- **Cloud hosting/SaaS:** CMS vendors are offering the software as a service through cloud hosting models.
- **CMS-based applications:** Many of the applications such as knowledge management, digital marketing applications, content-rich microsites, document repositories, and asset management systems heavily use CMS.
- **Dynamic components:** CMS platforms are increasingly using dynamic component model which can be managed without server restart. Open Service Gateway Initiative (OSGi) is one of the most popular component models used for this purpose

Various Roles in CMS

Digital content systems are accessed by various users' roles. Roles help in providing fine-grained access control to various CMS functions. Various user roles within a content management system are presented in Table 3.2.

Other roles include workflow administrator, publisher, legal reviewer, marketing reviewer, and others based on business needs.

3.2 CMS KEY DESIGN PRINCIPLES

Design principles form the guidelines for the CMS programs. The design principles can be used as reference during CMS development.

Listed below are main design principles, representing proven best practices adopted in CMS programs.

Reusability and Flexibility

While designing CMS components such as authoring templates, presentation templates, and workflows, we need to ensure that they are flexible enough to be reused across various scenarios and content types. We can enhance reusability by adopting open standards during component development and integration. Reusable content components greatly reduce content development effort and deliver high-quality content. We have discussed some aspects of reusability in Chapter 2 and we will touch base on template reusability in Chapter 5.

Table 3.2 Content Roles and Responsibilities

Content Role	Role Responsibilities
Content Administrator	• Sets up the sites • Assigns all the users their access rights • Manages and publishes content. • Has flexibility when it comes to creating a template. • Accesses content created by other content authors • Maintains the security settings of the site. The privileges for this role include administering roles, workflow, lifecycles, categories, and content types. • Performs the task of administering the overall content of the Web site • Assigns roles and permissions for various CMS resources • Manages and assigns permissions to the workflows
Content Author/Editor	Creates as well as edits and deletes content. A content author will get his/her contents published only after the created content has been reviewed by the respective content reviewer. Authors use pre-created templates and various tools offered by the CMS such as Rich Text Editor (RTE) image picker to author the content.
Content Reviewer	Reviews content created by content author before it is published over the Web site. Based on the performed review, the content reviewer can either approve or reject content. Content reviewer can use the preview functionality offered by the CMS on staging environment to review the overall look and feel.
Public Users	Public users are end-users who browse through the site content and perform actions. They can contribute user-generated content (UGC) through blogs, articles, wiki posts, and the like.

Taxonomy and Metadata Definition

Enterprise-wide taxonomy should be defined and integrated with CMS. Taxonomy provides the business terms and hierarchy that can be used as metadata for tagging content. Taxonomy can provide definition for various business terms. Taxonomy governance may provide consistent definition of metadata terms to CMS.

Taxonomy and metadata are discussed in more detail in chapter 6.

Standards Definition

Content architects should design standards used for managing content. Open standards should be given preference while storing, publishing, migrating, sharing, and integrating content. Some of the popular content standards include XML, DITA, JSON, and SCORM. CMIS (Content Management Interoperability Services) and JCR in content integration space. REST, SOAP, and ATOM/RSS are standards commonly used for content services, content publishing and feed-based content integration.

Using standards in the content processes provides greater flexibility, reusability, and interoperability and facilitates easier content integration. Standards-based systems can communicate and interoperate efficiently. Adopting open standards will eventually reduce the effort and total cost of program ownership.

Content standards may also help in exposing content as services and feeds to external systems. CMS should be configured to structure and format the content as per the defined standards.

A detailed discussion of content standards can be found in Chapter 7.

Consolidated Content Repository

Siloed content is one of the main challenges with enterprise content. The most immediate job after selecting a CMS is to consolidate enterprise content into a single repository. Enterprise content needs to be migrated or integrated into the new content repository. Enterprise CMS should provide a single view of content to all consuming applications. This will eliminate content duplicates and effort redundancy. Once content is aggregated, it is easy to provide consistent classification and metadata tagging to such content.

Governance Model

Efficient content governance model plays a key role in ensuring that content processes are aligned with business goals. It defines a clear role-responsibility matrix and defines standards and processes to efficiently manage content. Content governance provides a stable foundation for implementing content strategy and content road map.

Content governance has been discussed in detail in Chapter 2.

Content Syndication and Services

CMS should be configured to syndicate content to external systems through feeds, services, and as files. Content can be provided fully as a page or partially in the form of chunks. This will enhance the usability of the content.

For a detailed discussion of content services, see Chapter 7.

CMS Evaluation

We need to have a robust CMS evaluation methodology considering the business functionality fitment, product alignment with the enterprise road map, standards compliance, operational capabilities, and other factors. As CMS is a long-term investment, there should be a careful, multifaceted product evaluation process to finalize the CMS product.

Factors important for CMS evaluation are presented in Chapter 9 and Appendix C.

3.3 CMS CAPABILITIES AND ATTRIBUTES

These core capabilities form the key evaluation criteria during CMS evaluation and selection and are used for business functionality fitment. In this section we look at the core capabilities, key functionality, and CMS attributes that play a crucial role in CMS selection.

Desired Core Capabilities of CMS

Listed below are the main capabilities desired of a CMS:

- Flexible template: CMS should provide built-in authoring and presentation templates for creating content. Administrators should also be able to create custom templates to suit content needs.
- Metadata support: CMS should support built-in metadata support or should support easy integration with external metadata systems. The metadata should be used for tagging content.
- Content services: CMS should be able to expose the content as services and feeds in standard format. Common content services include content retrieval, content search, content update/creation, and content deletion.
- Standards compliance: CMS should be compliant to industry standards such as CMIS, JCR, and others so that it would be easy to integrate and exchange content with other systems.
- Flexible publishing options: CMS should be able to publish core content in various formats (XML, HTML, JSON and such) to various publishing targets (Web server, file system, service, etc.)
- Preview feature: CMS should support content and page preview feature. This helps reviewers preview entire content before approval.
- Business-friendly administration: CMS should provide easy-to-use interface to help administrators perform various administration activities.
- Versioning and archival: CMS should provide content check-in/checkout and versioning features. CMS should support content archival for specified time duration.
- Workflow: Content workflow should provide collaborative authoring, reviewing, and publishing. Workflow administrators should be able to configure skill-based routing rules for content review and event-triggered notifications.
- Integrators and adaptors: CMS should provide/support extensible plugins and adaptors to easily integrate with enterprise systems (such as enterprise portals, document repositories, asset repositories, taxonomy/metadata systems, etc.)
- Content search: CMS should provide basic content search and should be easily integrated with enterprise search engines. Content metadata, content types, content categories, and other similar content attributes should form the basis for content search.

- Collaborative content: CMS should be easily customized/configured to build collaborative platforms such as blogs, wikis, forums, document sharing platforms and such.
- Content analytics and reporting: CMS should provide basic content analytics and report content usage metrics.
- Disaster recovery (DR) and business continuity: CMS infrastructure should facilitate development of DR site through content synchronization jobs.
- Content access and permission model: CMS should provide robust permission model, access control list (ACL), and role-based access to content resources (such as templates, workflows, admin pages, etc.). CMS should also be easily integrated with LDAP and similar user registries.
- Metadata-based security: CMS should provide security metadata that can be used to secure content and its associated resources. User role and access permission are some of the examples of security metadata.
- Cloud support: Modern-day CMS should support cloud hosting and cloud deployment model.

The key technological capabilities of a digital content management system is given in the Figure 3.2:

Content services: This layer includes the content services exposed through SOAP and REST services for external systems. Content services provide functionality to view, update, and delete content. Additionally, personalization, content feeds, and content widget can also be provided to enable content plug-and-play model.

Core content components: This is the heart of the CMS, providing the main content capabilities. This includes various modules and components such as reusable content templates, digital asset management (DAM) modules, content repository for storing Web content, metadata model to store the metadata hierarchy and making it available to tag with Web content during authoring and publishing, various content workflows for authoring, reviewing, publishing, and localizing content, content optimized for multiple devices, content-based collaborative platforms (such as knowledge management systems), content localization modules, content transformation tools (such as XSLT), campaign content, content preview that allows the final reviewers to preview the final content, content administration, content security through role-based content, and functionality access.

Content integration components: This module helps CMS interact with essential external systems such as enterprise portals, Web analytics, translation management systems, asset management system, document management system, external social media, SEO, and other enterprise interfaces. Integration happens through APIs, connectors, services, or using CMIS. Content syndication service may help in syndicating content from various sources.

Enterprise content sources: This includes all enterprise interfaces that will be used by CMS, such as metadata management systems (MMS), other CMS systems, enterprise database, system-of-record (SOR) such as a pricing system, asset servers

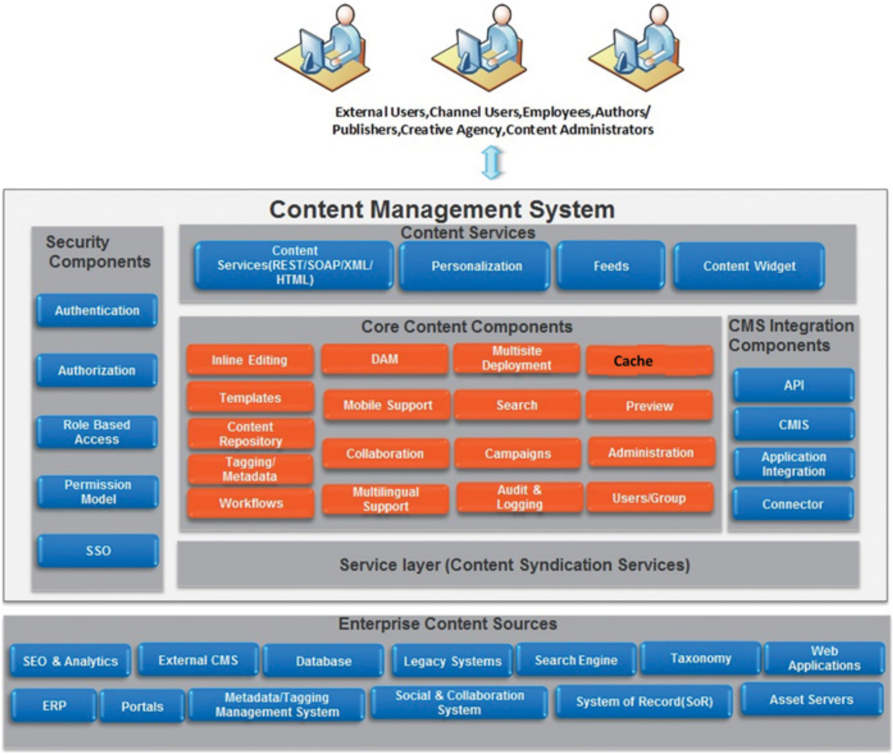


Figure 3.2 Content Management Capability Model

for digital assets, and social engagement and collaboration systems to obtain the collaborative content and search engine.

Security components: CMS also provide in-built security features such as authentication (native or LDAP integrated), authorization (through fine-grained permission model), role based access to content and functions, single-sign-on (SSO), and so on. Metadata module can also be used as security component as it can be used to filter the access based on tagged values.

Content governance defines content processes and defines the roles and responsibilities for managing the content. This includes key content management roles such as, among others, content authors, content publishers, and content administrators. We have seen various content roles in an earlier section.

Main Functionality of Core CMS components

Let’s take a deeper look at the key functionality of core CMS components we discussed earlier. We will look at the detailed design of these core components in Chapter 4.

Content creation, edit, and preview

This is the core capability of any CMS. The system should provide a user-friendly content editing and creation interface. Most CMS provide rich text editors (RTE) and WYSIWYG (What you see is what you get) editors and allows users to tag metadata, attach files, graphics, and videos, and specify the content language.

The structure of content largely depends on the content-authoring templates.

Once content is authored, authors and editors need to preview it to check for formatting, layout, and overall structure. Some CMS also provide a feature known as in-line editing, with which authors can edit content in preview mode.

Content versioning

Content versioning is one of key requirements of a content management system, as it helps in rollback and content recovery. A version is a snapshot of content at a given time along with all the associated metadata, content category, assets, tags, keywords and annotations. Localized content has its own versions and is not dependent on English master copies. Normally, as part of content publishing, a content version will be created to store the copy of the published version for future references.

The content versioning feature keeps the historical copies of content and assets and help us in creating the release.

Content workflows

Workflows consist of a series of steps and are closely modeled on underlying business processes. Multi-step approvals, reviews, business rules implementation can all be achieved through workflows. Some of the most commonly used workflows are related to content publishing, content authoring, and content translation.

Content archival

Business and regulatory requirements require archiving and versioning content. CMS will provide built-in tools for achieving these.

Templates

Templates provide structure and layout for the content fragment and page. There are mainly two kinds of templates: content authoring template and content presentation template. Authors and editors can use authoring template for creating content, and presentation templates will be used to format the presentation.

Content analytics

CMS should also support adding Web analytics–related code/script during authoring. This can be part of content or can be provided as a separated element in the content template. This helps in tracking content usage, thereby providing insights into effectiveness of the authored content.

Content performance

CMS support content performance through various means such as page caching, content fragment/chunk caching, etc. Typically cached content will be invalidated when the new content is published through content publishing workflow.

Digital asset management

Some content management systems also contain sophisticated digital asset management (DAM) modules that provide asset management capabilities such as asset creation, uploading, editing, association, tagging, asset resizing, and so on. In some other scenarios an enterprise may be having a centralized asset management system and CMS needs to be integrated with it. In such cases CMS will be having asset management workflows.

Document management

Content systems can also be used as document repositories. CMS provides many built-in features to manage the lifecycle of the document, including, among others, document uploading, document tagging, document searching, and document archival. Alternatively, CMS can also be integrated with enterprise document management systems.

Content publisher

This is another core component of CMS wherein the reviewed content will be published to the destination. The destination could be a Web server, a file system, a database, or it could be exposed as a Web service.

Content taxonomy

Content is normally organized in a logical hierarchy for easier management. CMS should support this logical grouping of content nodes and support the content taxonomy.

Desired Attributes of CMS

The core attributes required of any CMS are presented below. These attributes can be used in selection of a CMS product or to fill the essential gaps in existing CMS systems.

Ease of content creation

- CMS should support a single, centralized repository that may help the organization have content contributors all over the world, who may be feeding content into the one centralized system. This will help in gathering the latest information from across the globe in one location.

- CMS should possess a standard WYSIWYG (What You See Is What You Get) editing tool for editors, which can be easily accessed and operated by non-expert users.
- CMS should support platform-independent content.
- CMS should have the facility so that the user can view content, which he/she drafted, as it will appear on the live site.
- CMS should expose the content service so that the users outside of the local network can access/edit the content.

Efficient content management

- CMS should have a workflow-managed system of content control, from creation to archival, through user roles and permissions. Workflows can be customized to model the organization's business processes and provide task notifications and alerts to users when content moves through the workflow.
- CMS should possess a strong authentication and authorization mechanism to ensure that a user is able to access, change, and approve content solely based on his/her role in the organization.
- CMS should have a version control and archiving mechanism so that the changes to content can be tracked and archived for legal accountability, and accessed through an interface.
- CMS should automatically populate metadata fields using existing data, and require compulsory metadata fields when creating content.
- There should be a mechanism so that a piece of content could be used in multiple locations on a Web site, or syndicated to sites outside of the CMS repository via RSS.
- CMS should produce user-friendly URLs for dynamically generated Web pages. The system manages changed links to other pages within the CMS.
- CMS should have a provision to generate a large range of reports containing data about CMS content and site users as well as the Web site's end-users. It should support various data export formats such as, among others, XML, CSV, PDF, Text, and MS Excel.
- CMS should support content localization through translation workflows.

Efficient content publication

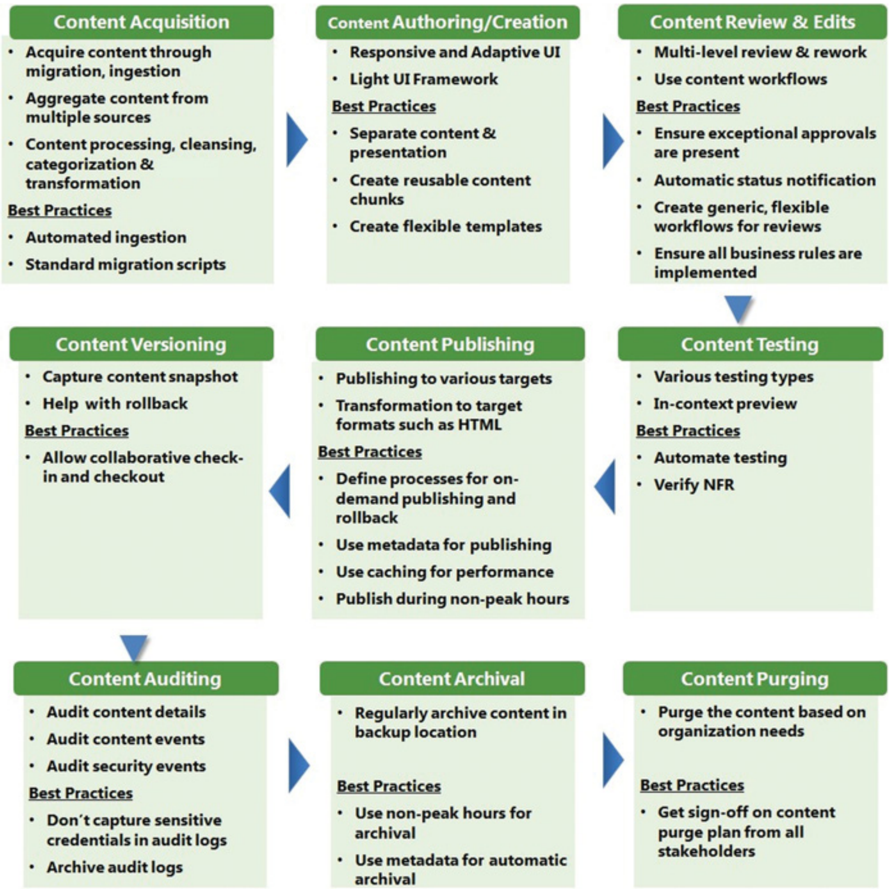
- Design of pages is controlled using presentation templates that provide the layout, look, and feel for the site. Templates are usually developed as flexible, reusable structures and they act as containers for content.

- A CMS should aid in publishing accessible Web pages and provide various features such as metadata publishing, distributed publishing, and publishing in various formats.

After having seen the capabilities and attributes of CMS, we can now discuss the content management capabilities of CMS.

3.4 CONTENT LIFECYCLE MANAGEMENT IN CMS

In order to manage the entire lifecycle of enterprise content, CMS provides native features, built-in workflows, and tools in all these stages of content. In this section we first take a look at all content lifecycle phases, after which we can discuss the role of CMS in each of those phases. Various steps of content lifecycle are depicted in the Figure 3.3. For each of the content lifecycle phases we have also listed the best practices that can be followed in that phase.



Content Acquisition

- Acquire content through migration, ingestion
- Aggregate content from multiple sources
- Content processing, cleansing, categorization & transformation

Best Practices

- Automated ingestion
- Standard migration scripts

Content Authoring/Creation

- Responsive and Adaptive UI
- Light UI Framework

Best Practices

- Separate content & presentation
- Create reusable content chunks
- Create flexible templates

Content Review & Edits

- Multi-level review & rework
- Use content workflows

Best Practices

- Ensure exceptional approvals are present
- Automatic status notification
- Create generic, flexible workflows for reviews
- Ensure all business rules are implemented

Content Versioning

- Capture content snapshot
- Help with rollback

Best Practices

- Allow collaborative check-in and checkout

Content Publishing

- Publishing to various targets
- Transformation to target formats such as HTML

Best Practices

- Define processes for on-demand publishing and rollback
- Use metadata for publishing
- Use caching for performance
- Publish during non-peak hours

Content Testing

- Various testing types
- In-context preview

Best Practices

- Automate testing
- Verify NFR

Content Auditing

- Audit content details
- Audit content events
- Audit security events

Best Practices

- Don't capture sensitive credentials in audit logs
- Archive audit logs

Content Archival

- Regularly archive content in backup location

Best Practices

- Use non-peak hours for archival
- Use metadata for automatic archival

Content Purging

- Purge the content based on organization needs

Best Practices

- Get sign-off on content purge plan from all stakeholders

Figure 3.3 Content Lifecycle Stages and Best Practices

Content acquisition and processing: Content can be acquired through various means: content migration, content authoring, user-generated content (such as blog, wiki, article), content syndication, pulling external feeds, etc. For creating marketing content, authors would get the master content along with associated assets from digital marketing team. Other forms of content acquisition include content aggregation, content ingestion, content conversion, and content migration. After content is acquired, it is processed, which includes content cleansing, content categorization, content transformation, and other activities that lead to its being converted into a structured format. This process helps in better search indexing and content retrieval.

CMS should be capable of ingesting content from different sources, applying standard templates, and publishing it.

Content authoring and creation: Authors create content as per business requirements using authoring templates. Content can also be created through content templates, which are equipped with user-friendly RTE and WYSIWYG editors. Templates are one of the core features of CMS that can be used for content creation.

Content review, editing, and approval: Authored content is reviewed and approved by relevant stakeholders. Based on review comments, content is updated to ensure consistency and adherence to visual guidelines. We can leverage CMS's built-in content workflows for review and multi-step approval.

Content testing: Content issues could be validated in preview mode and can be re-edited to address the content defects. We would also carry out various functional and non-functional testing. Few content systems provide support for A/B testing and other user testing types.

Content publishing: Content will be published to the publishing destination such as file systems, servers, databases and content repositories. Usually content and asset publishing happens using publishing queues that help in batch publishing. Once all the content and related assets are reviewed and approved, and ready for publishing, the workflow will move to publishing step (or sometimes content publisher will initiate the new publishing workflow). The publishing step will do the following:

- Transforms the core content into presentation format such as HTML needed for target environment (such as file system or web server). This is achieved through XSL transformation or by using presentation templates
- Content will also be transformed to XML or JSON format based on needs of target environment.
- Once all the presentation files are ready, the publisher will publish the files along with dependent assets like JS/CSS/images/media files, metadata etc. to the target location
- Publishing normally happens in batch mode for optimization purposes.

All CMS provide publishing mechanism through various in-built features such as publishing workflows, multi-site publication and through synchronization of author and publish instances.

Sometimes publishing process is also referred to as content delivery. Content Delivery mainly includes publishing the content in various formats. Publishing can simply include copying the reviewed content to the presentation server or it could also include exposing the content as services to up stream systems. Content presentation process normally includes publishing workflows, presentation transformation and such activities

Content versioning: Content will be versioned at regular intervals. The snapshot of content helps in content rollback and provides a checkpoint for content releases. The functionality includes content check-in/check-out, content differences, content preview; content locks and allows collaborative content management. Most CMS provide in-built content versioning features which could be leveraged for this.

Content auditing, archival and purging: Content auditing is mainly used to track user changes to the content and assets. Auditing process would capture values such as user name, role, asset/content id, Created date, updated timestamp and such details. Content will be archived and purged based on organization and regulation needs. Archived content preserves the know-how and knowledge for future purposes and complies with legal regulations. Stored knowledge can also be used for training purposes. Once the content reaches its end-of-life it will be either purged or archived based on business and legal requirements. Here again, we can leverage in-built CMS features for auditing, archiving and purging content.

So far we have seen the mean features of CMS including its capabilities, attributes and functionality. Let us have a brief discussion about some of the popular open source CMS products.

3.5 A BRIEF DESCRIPTION OF OPEN SOURCE CMS AND JCR

Let us have a quick look at some of the popular open source CMS systems. In coming chapters we will provide examples from these CMS platforms while explaining the concepts.

We will also look at the details of Apache JackRabbit, a JCR implementation. As JCR is one of the popular content repository standards, we will have a deep-dive into the JCR implementation details

Drupal (https://www.drupal.org/)

It is one of the most popular Open source content management systems. Developed in PHP, Drupal helps users in organizing information and to publish content as websites. The platform provides a modular framework on which we can build custom modules. Drupal provides a taxonomy hierarchy that can be used for keyword tagging.

Drupal technology stack can be categorized into these groups

- Core modules: These modules provide the basic functionality and can be extended. Core modules include templates, Pages, blogs, comments, forums, polls, search, logging, feeds, menus, caching, security, taxonomy, user profile, user management, access control etc. Core content types are article, basic page, Blog entry, Book page, comment, forum, and poll and custom types.

- Nodes: Content will be stored as nodes in nodes table along with content type.
- Extensions can be implemented using hooks and callbacks to extend the default platform behavior.
- Themes can be used to customize the look and feel of the application. Themes are made up of CSS files, XHTML, PHP variables. The core building blocks of themes are regions, blocks and menus.
- Localization: Drupal provides support for more than 100 languages

Drupal has large community base who has contributed thousands of modules

Drupal can be used as CMS, knowledge management platform, collaboration platform (blogs, forums, and community website), publication platform and as a website or a portal. Drupal can be best used for building a complex and massive websites requiring large data handling, huge content volumes, large user base and transaction handling.

Joomla (https://www.joomla.org/)

Joomla is another popular open-source CMS built on PHP. The platform makes use of object oriented programming techniques and provides easy theming and flexible administration features. We can use Joomla extension and plugins built on Joomla Application Framework to build sophisticated features. Joomla templates can be used to build the page layout and its look and feel. Joomla plugins can be used to build professional website very quickly. Joomla provides additional features such as pages, polls, article, news feed and others.

Joomla is widely used to build intranet websites, standard content-based web sites, portals, publication sites, technology forums, e-commerce platforms, collaboration platform, finance applications and such.

WordPress (https://wordpress.org/)

This PHP-based CMS is used mainly for creating blogs and website. The intuitive user interface helps administrators to quickly build the web sites with minimal technical assistance. WordPress provides many in-built blog management features such as blog moderation, notification, posts, categories, formatting and others that help users to quickly setup and create blogs.

The user interface mainly consists of administration panels, dashboards and features to add/edit/manage posts. We can customize the appearance using themes, widgets, and headers.

Plugins can be used to extend the default functionality. User community has contributed over 40,000 plugins for WordPress. WordPress is most effective in building simple web sites, blogging sites for everyday use.

JCR Implementation: Apache Jackrabbit

Java content repository API (JSR-170 and JSR-283) specifies standards to access a complaint content repository using the classes and interfaces in javax.jcr package. Apache Jackrabbit is one of the most popular implementations of JCR standard. JCR defines "Compliance Level" which specifies the features supported. Various compliance levels are given below:

- Level1: A repository is called a Level1 complaint if it is a read only repository
- Level2: A repository is called a Level2 complaint if it is a writable repository

Besides these compliance levels, the content repository can also provide features such as Versioning, Transactions management, Search, Locking and Content Observation, Security.

Apache Jackrabbit is a Level 2 complaint, which also implements many advanced features mentioned above. In the Jackrabbit repository we can store variety of structured content (such as product information data) and unstructured content (such as scanned files, binary files).

The data model mainly consists of workspaces, which are accessed from user sessions. Each workspace has a unique name and a root node with multiple child nodes. The data in each workspace is stored through nodes and its properties. Each node has one parent and contains multiple child nodes and multiple child properties. We can use a path to access Node and its property. Properties can be single or multivalued and they are of type String, Boolean, integer, binary etc. Once the user logs in, he/she can perform CRUD (Create, Read, Update, and Delete) operations on the workspace data.

Nodes can be of two categories: Primary and Mixin. Primary node defines the domain node structure and mixin node defines the metadata of that node. For example, while defining the node for storing the product content details, the primary node would be *demo: productDetails* with properties such as *demo:productId, demo:productName, demo:productCategory* and the mixin specifies node functionality such as *mixin:lockable, mixin:versionable* and such

We are going to discuss architecture concepts of Apache Jackrabbit in chapter 4

3.6 CHAPTER SUMMARY

- This chapter mainly discusses the basics of digital content management such as CMS definition, design principles, capabilities, attributes etc.
- Content Management System is a set of technologies and processes, which helps in creating, publishing, customizing, and management of all content across an enterprise/division for managing end-to-end lifecycle along with its dependencies.
- The main business drivers for CMS are Efficient content management, Agile and fast processes, Management of Information explosion, Content

consolidation, Optimal user experience, Streamlining information updates, Content security, Information Quality, Return on Investment

- The key challenges during CMS implementation are Content structure variations,Existence of content silos, Content migration challenges, Challenges in modeling Metadata hierarchy, Challenges in defining Site taxonomy, Modeling business processes, Publishing delays, Omni-channel content delivery, Content performance
- Content Administrator, Content Authors/Editors, Content Reviewer, public users are some of the common content roles
- The key CMS Design Principles are Taxonomy and metadata definition, Standards definition, Consolidated content repository, Governance Model, Content syndication and services, CMS Evaluation
- Core Capabilities of CMS are Flexible template, Metadata support, Content services, Standards compliance, Flexible publishing options, Preview feature, Business-friendly administration, Versioning and archival, Workflow, Integrators and adaptors, Content search, Collaborative content, Content analytics and reporting, Disaster Recovery (DR) and Business continuity, Content security and Cloud support
- CMS capability model include content services, core content components (such as templates, repository, metadata, workflow, DAM, search, released access, preview etc.), integration components
- The desired attributes of CMS are Ease of content creation, Efficient Management of the content and Efficient Content Publication
- Content Lifecycle management in CMS includes Content acquisition, Content processing, Content management, Content presentation and content retention
- Drupal, WordPress, Joomla are some of the popular CMS products
- Architecture details of Jackrabbit which is a JCR implementation is explained.

Chapter 4

Content Management System Architecture

CMS provides a robust ecosystem and implementation platform for efficient realization of content strategy. We have explored the core CMS components, features, and attributes in previous chapters. CMS provides various built-in components such as templates, workflows, multi-site management, localization, and publishing, which can be used for end-to-end content management. This chapter provides an in-depth discussion of core CMS components.

In this chapter we will look at architecture elements of a content management system. We will extend the concepts discussed in earlier chapters and have a deeper look into various CMS components and their roles in content strategy. We start by defining various phases of CMS implementation approach. We then discuss modern CMS architecture patterns including MVC, SOA, n-tier, and Microservices architectures. We then look at CMS value articulation framework and the key CMS solution design principles. We will have an in-depth look into design aspects of CMS solution components such as multi-site management, localization, content folders, content URLs, CMS infrastructure, and CMS maintenance. Finally we discuss various CMS reference architecture by looking at CXP architecture, knowledge management architecture, digital marketing platforms based on CMS, and Apache Jackrabbit architecture.

CMS architects, content architects, content strategists, CMS developers, and enterprise architects will find this chapter useful.

4.1 CMS DESIGN AND ARCHITECTURE

A CMS ecosystem consists of components to process and manage Web content, documents, images, XML documents, assets, etc. This section provides an overview of CMS design concepts for an enterprise. As a first step, we will look at the steps and methodology involved in a robust CMS implementation.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

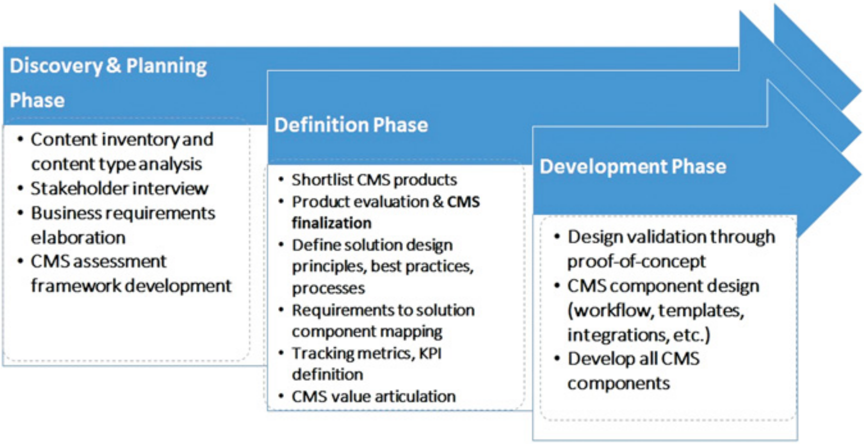

Discovery & Planning Phase
• Content inventory and content type analysis
• Stakeholder interview
• Business requirements elaboration
• CMS assessment framework development
Definition Phase
• Shortlist CMS products
• Product evaluation & CMS finalization
• Define solution design principles, best practices, processes
• Requirements to solution component mapping
• Tracking metrics, KPI definition
• CMS value articulation
Development Phase
• Design validation through proof-of-concept
• CMS component design (workflow, templates, integrations, etc.)
• Develop all CMS components

Figure 4.1 CMS Selection and Implementation Methodology

CMS Implementation Approach

An enterprise has to carefully evaluate the need for the CMS and use it efficiently to address existing gaps and to fulfill organization goals. We need to evaluate candidate CMS products and select the most appropriate product to fulfill organization goals. Once CMS is finalized, we need to develop the solution using the CMS (referred to as “CMS implementation”).

Figure 4.1 offers an overview of the CMS selection and implementation methodology. The methodology details three phases along with best practices and steps.

A new CMS selection and implementation methodology consists of these phases:

Discovery and Planning Phase: Compile and analyze the list of all essential content and assets requirements through interviews with various business stakeholders to understand business needs and drivers. Use the content assessment questionnaire and other requirements-gathering techniques to capture the general content requirements. Business function requirements (such as digital marketing–related requirements) also need to be captured. Some of the planning-related questions are:

- What are the challenges in existing systems?
- What are the prominent user journeys?
- Are there any content localization requirements?
- What are SLAs related to scalability and performance?
- What is the existing content creation and publishing process?

Elaborate the content and CMS-related requirements. We have already discussed some of the content requirements elaboration methods in Chapter 2, which can be used in this scenario.

Definition Phase: Compile the responses from the questionnaire and evaluate all the candidate CMS products. Use an exhaustive product evaluation framework and finalize the CMS. CMS architects should define the CMS solution design principles (detailed in the next section), CMS best practices, CMS processes, and CMS management guidelines. A CMS architect should also map the business goals and requirements to CMS solution components and define metrics to articulate the CMS business value. We will look at these topics in the next section.

Note: We have detailed an elaborate CMS evaluation framework in Chapter 9. A comprehensive CMS evaluation template is provided in Appendix C.

Development Phase: Gather detailed business requirements and use PoC-based feasibility assessment technique to understand all the CMS customizations/extensions/configurations needed to implement the specified requirements. Develop a detailed design for all CMS components (such as templates, workflows, site creation, deployment, etc.). Develop the CMS components iteratively.

Once the CMS is fully set up and customized, we can start developing more specific CMS components such as templates, workflow, a multi-site management module, DAM integration, etc.

The first best step in architecting any system is to have a good understanding of applicable architecture patterns for the applicable solution domain. We can use this knowledge to identify the most appropriate pattern or create a variant of it to suit the current scenario. Let us look at some of the most popular patterns in the context of a CMS-based solution.

4.2 MODERN CMS ARCHITECTURE PATTERNS

In this section we discuss the main architecture patterns used in CMS solutions. In most of our case studies we have discussed layered MVC architecture, n-tier architecture, and service-oriented architecture. Hence we will briefly discuss these two architecture patterns, and elaborate on Microservices architecture.

Note: The architecture patterns are discussed in the context of content management scenario. Hence examples are mainly focused on content.

MVC Architecture

More often than not, CMS applications are Internet-facing B2C sites. MVC is one of the most popular and proven architecture pattern for Internet applications. Due to its wide acceptance and popularity, there are numerous open-source frameworks (such as Apache Struts, Spring MVC, etc.) that are built using this pattern.

Model view controller (MVC) architecture pattern neatly separates concerns into various layers. It abstracts each layer from other layers, and interaction happens through well-defined service contracts. In MVC architecture solution components would be partitioned into separate layers (each with distinct responsibility), providing layer-wise scalability and better maintenance. The architecture pattern provides clean separation of layers and their responsibilities. The components in each layer will access logic and resources in other layers through well-abstracted interfaces that enable virtualization. For example, the view layer will be completely abstracted from the model layer and the controller layer.

In a generic MVC architecture, a view layer consists of presentation components (such as HTML, XML, taglibs, etc.), which render the end-user interface; controller mainly handles the routing and navigation logic to find the most appropriate model component to handle a given request and selects the view to render the data obtained from model layer components.

It is important to understand the difference between a complete Web application and the traditional use case for CMS for applying MVC pattern to CMS solution. In a typical Web application, we have various modules to handle distinct responsibility for an end-to-end flow; hence we can design modules for each of MVC layers. In a CMS solution scenario, on the other hand, CMS is used mainly as content repository. In this setup CMS mainly serves page content and it is be "used" by an enterprise application (such as portals or CXP). In such scenarios the role of CMS would be mostly limited to model layer.

As discussed in previous chapters, modern CMS platforms are converging with presentation platforms, and the line between portals and CMS is blurring. Many enterprises are using CMS portals that provide end-to-end components required for the solution. This elevates the role of CMS, and we can identify various modules in a CMS portal for implementing MVC architecture.

In the CMS portal scenario, a model layer consists of content repository, data persistence components, and other back-end services (such as translation services, database services, metadata services, etc.); a view layer handles content presentation concerns (such as templates, pages, content widgets, page layouts, themes, etc.); and the controller handles routing logic based on the requested user action (normally implemented by routing servlets). In order to fully implement an MVC pattern for a CMS portal, CMS portal's view layer components should normally use service calls (mainly REST calls) to model layer components. We also need a standard format of data (such as XML data) containing core data without HTML elements.

Benefits of MVC pattern for CMS application

Decomposition into appropriate MVC layers provides the following benefits:

- Loosely coupled content modules: Since services are used for fetching content, model layer can serve the content form native content repository or can get content from a remote repository. This enhances flexibility and extensibility of the content modules and enables us to replace current repositories in the future, if needed.
- Separation of concerns: Like in a typical MVC architecture, the view layer in a CMS portal handles the presentation logic by injecting presentation elements into core content; the model layer sends only core content in a standard format.
- Abstraction of content modules: Modules and their inner structure in each of the layers are abstracted from other layers.

Table 4.1 CMS components in MVC layers

Layer	CMS components
Model	Content services, content repository, metadata service, taxonomy service, database component, OSGI bundles
View	CMS portal pages, responsive themes, page layout, presentation libraries, page templates, navigation modules, widgets, portlets, multilingual resource bundles
Controller	Routing servlet, dispatchers

- Ease of testing: We can independently test content modules in each layer by stubbing other layers to enhance quality of the deliverables.
- Robust integration: Due to contract-based services, we can easily integrate CMS modules with external systems and services.

CMS components for MVC layers

Most CMS products provide built-in modules for each of the MVC layers, and developers can use them or extended them to build a modular application. We are seeing an increasing convergence of CMS and portals, and hence many CMS products are also offering good presentation engines bundled along with regular CMS components. Table 4.1 is an overview mapping of CMS components in an MVC architecture.

We will look at this architecture pattern in many of the case studies discussed in this book.

N-tier Architecture and N-layer Architecture

In tiered architecture, components are logically categorized into physically separated application tiers, with each tier handling a distinct responsibility. A typical three-tier architecture consists of a presentation tier (contains components that handle user interaction), an application tier (contains components that handle business logic), and a database tier (contains components that handle persistence logic) for handling tier-specific responsibilities.

As the complexity of the application increases, other tiers may be present, such as content tier, integration tier, collaboration tier, orchestration tier, services tier, middleware tier, etc. This leads to N-tier architecture.

Similar to MVC architecture, N-tier architecture handles “separation of concerns” across various tiers to provide easier maintainability and higher scalability and availability. Individual tiers can be sized based on the anticipated load and performance requirements, which offers optimal scalability. Each individual tier can also be managed independently through tier-specific security configuration and developing modular components with high reusability.

In N-tier architecture, a CMS system is hosted in its own CMS tier, preferably in a clustered configuration. The tier is mainly responsible for handling content and asset concerns.

N-layered architecture provides logical separation of distinct layers. Presentation layer, business layer, and data layer handle layer-specific responsibilities, and all the logical layers may be hosted on same hardware. For instance, a three-layered architecture of Drupal CMS application consists of MySQL database in a database layer, custom modules in a business layer, and PHP, themes, views, and HTML in a presentation layer.

Challenges and best practices in implementing layered architecture in CMS applications

Few CMS products would not provide cleanly separated layers. Presentation components allow the developers to directly invoke database calls, bypassing the business logic layer. Due to this anti-pattern, solutions developed in such scenarios tend to be tightly interdependent, making it difficult to extend them or make changes in the future.

We can overcome this challenge by enforcing the technology governance structure. CMS architects can design the application framework with application components and APIs for each layer. This abstraction should strictly enforce the loose coupling across layers and restrict any communications that bypass the layer. For instance, presentation layer components (such as JSP, tag libraries, and widgets) should only communicate with business objects in a business layer. Business objects in turn would invoke corresponding data sources. Application developers can build the solution on top of this framework. This would help build a neatly separated layered architecture.

With a rare exception of a few legacy CMS systems, most modern CMS platforms provide built-in support for layered architecture. CMS architects need to analyze the layering support offered by CMS products and address the gaps (using application framework components) wherever required.

Service-Oriented Architecture (SOA)

Service-oriented architecture consists of loosely coupled, self-contained services. Services are typically independent, stateless, and encapsulate application functionality. Service orientation is widely established as an approach to achieve agility, reuse, resource virtualization, and interoperability. Adopting the key principles of SOA ensures architecture goals of interoperability, upgradeability, and manageability. Service orientation also makes it easier to scale individual services based on the demand, thereby improving overall scalability. In addition, service orientation enables virtualization.

Description

The design of individual components will adopt key service-oriented principles of encapsulation, loose coupling, and contract-based integration. Wherever possible, the service interface is designed based on established industry standards.

The services are stateless in order to enable virtualization; wherever needed, the state is managed by state management services that virtualize the use of resources for state information, which helps in deploying the service in grid-based architecture.

In SOA, components in each layer interact with components in their subsequent layers through services. Services are designed to be modular, independent, and reusable so that it would be easy to extend their functionality.

In this architecture pattern, each layer exposes its core functionality as service. We could logically segregate overall solution into layered services. Typical service layers are presentation services layer (which provides presentation handling and rendering function, personalization), business services layer (which handles business logic), data services layer (which handles data and persistence logic), content services layer (which handles content management concerns), security services layer (which provides authentication, authorization, identity management, and user provisioning services), infrastructure services layer (which provides services related to logging, caching, auditing, exception handling, and notification), and search services layer (which provides basic and advanced search features).

In complex scenarios, we may use message-oriented middleware (MoM) and enterprise service bus (ESB) to centralize service aggregation logic and to handle concerns such as service governance, data transformation, protocol transformation, routing, validation, security, monitoring, and others.

SOAP- and REST-based Web services are widely used across applications implementing SOA. XML and JSON are standard data exchange formats used in Web services. We will discuss SOAP, REST, XML, and JSON in more detail in the "Content Standards" section of Chapter 7.

In CMS scenarios, SOA is widely adopted for two scenarios:

- Enterprise service integration to integrate CMS with enterprise interface through services. CMS integrates with upstream systems such as database, translation management system (TMS), and metadata system through services.
- Exposing content services to downstream systems (such as portals, presentation engines, content aggregation engines, mashups, widgets, etc.). Main content services are:
    - Content retrieval service to fetch the content based on metadata or content identifier.
    - Update content service to update content in content repository
    - Delete content service to delete the specified content from content repository
    - Tagging service to add metadata tags for specified content
    - Publishing service to publish content to various destinations
    - Content search service to search for content based on query parameters
    - Workflow services to manage content workflows.

As the Web is the predominant consumer of CMS content, content services are largely exposed via REST-based Web service with JSON as the main data exchange format. For instance, Apache Sling framework exposes JCR content as REST services. Similarly, most CMS products expose content functions via Web services.

Though services represent the main form of interacting with CMS, few legacy CMS systems offer only API-based integration. With such scenarios we can service-enable those CMS systems by building a content services layer using CMS APIs. Content services layer abstract CMS APIs from clients, and it can handle other responsibilities such as caching, content transformation, and security, among others.

Microservices Architecture

Microservices architecture is an emerging trend in highly scalable applications. Generally speaking, a microservice architecture model is about development of a composite application through aggregation of smaller, independent services. Microservices architecture creates a decoupled model by breaking down monolithic application into smaller, modular, and independent services. Microservices offer high platform scalability and augment a distributed hosting and development model. Each function is broken to its granular level (based on "single responsibility principle"), while designing a microservice and each microservice manages its own data and infrastructure using lightweight communication model such as REST. Salient features of microservices architecture are listed below:

- **Independent development:** Developers can develop microservices faster and in parallel through independently buildable code base.
- **Independent deployment:** Unlike monolithic applications, there is no need to redeploy entire code base for individual component changes; each microservice can be independently redeployed. This provides a great DevOps advantage in release management and provides faster time to market.
- **Independent and faster QA:** With individual microservice development and testing, testing cycles and release cycles can be greatly reduced.
- **Technology independence:** Each microservice can use a different technology, and technologies/platforms can be switched more easily. This simplifies the adoption of newer technologies.
- **Decentralization:** Process management, content management, and data management and governance can be decentralized.

A simple example for an e-commerce Web page can be used for explaining microservices. In a monolithic application scenario, the product page user interface would invoke a server component to get the data related to product information, product recommendation, product reviews, and product discussion. In microservice architecture, each of these functions is represented by independent microservices, and the product page UI invokes four microservices in parallel to render the final presentation. In Figure 4.2, an API gateway is shown for better performance.

Product page UI invokes four microservices through API Gateway. The API gateway acts a centralized service invocation layer and adds other features such as caching, governance, etc.

Table 4.2 lists important distinction between SOA and microservices.

Microservices for content management system

This section discusses a number of methods to implement microservices architecture in a CMS scenario.

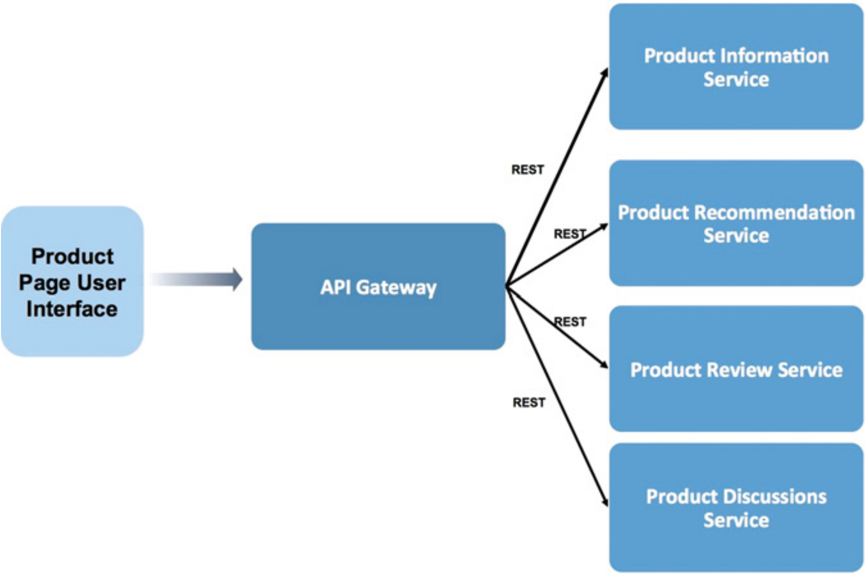

Product Page User Interface
API Gateway
REST
REST
REST
REST
Product Information Service
Product Recommendation Service
Product Review Service
Product Discussions Service

Figure 4.2 API Gateway-based Microservices architecture

Custom development of CMS microservices

In order to build microservice architecture in a content scenario, we first need to identify core content services that can be decoupled. Below are some of the main microservices:

- Authoring microservices: content model service, field validation service, content preview service, content storage service, etc.

Table 4.2 SOA vs. Microservices

	SOA	Microservice Architecture
Infrastructure	Uses shared infrastructure	Manages its own infrastructure
Scalability and availability	Depends on the underlying shared infrastructure and dependent components	Provides individual scalability and availability
Scope and granularity	Normally coarse-grained services with a broader scope. SOA architecture is mainly used for core enterprise services which do not change often. For instance SOA architecture is used for core banking services.	Fine-grained independent services. Microservices is used for digital platforms which need frequent changes due to agile development model.
Communication standard	SOAP with XML or REST with JSON	Mainly REST with JSON

- Publishing microservices: site publishing service, file publishing service, database publishing service, CMS publishing service, Web publishing service, etc.
- Content management services: content versioning service, content checkout service, etc.
- Search microservices: basic content search service, advanced content search service, metadata content search service, etc.

Each content microservice has its own independent process and can be independently maintained and deployed. In this scenario, we can use best-of-breed technology/framework for each of the microservices.

One of the natural complexities in this scenario is managing the interdependencies across various microservices and integration of various microservices to realize a full-fledge CMS feature.

Usage of CMF for developing microservices model

Content management framework (CMF) can be used to build a decoupled content management system through scalable and reusable functionality bundles. We can use these bundles as building blocks to develop the application with required CMS functionality or develop a general purpose CMS. Even in this case we can choose a best-of-breed bundle suitable for a particular functionality.

Use CMS plugins for implementing microservices architecture

We can leverage existing CMS plugins to develop a microservices model. For instance, we can use a microservices module in Drupal CMS to implement microservices architecture. We can develop services based on OSGI bundles to provide required functionality for JCR-based content.

4.3 CMS VALUE ARTICULATION AND SOLUTION PRINCIPLES

For the long-term success of CMS, business stakeholders should be able to accurately measure and appreciate the value derived from CMS. All concerned stakeholders should also be able to define and track the success metrics aligned with business goals. This helps them take course-corrective actions in case of any deviations. In the coming sections we will discuss the CMS value articulation framework and key design principles.

CMS Value Articulation Framework

While architecting a CMS solution, we need to understand the business vision and align the solution to fulfill the vision and use metrics-based measurement of solution

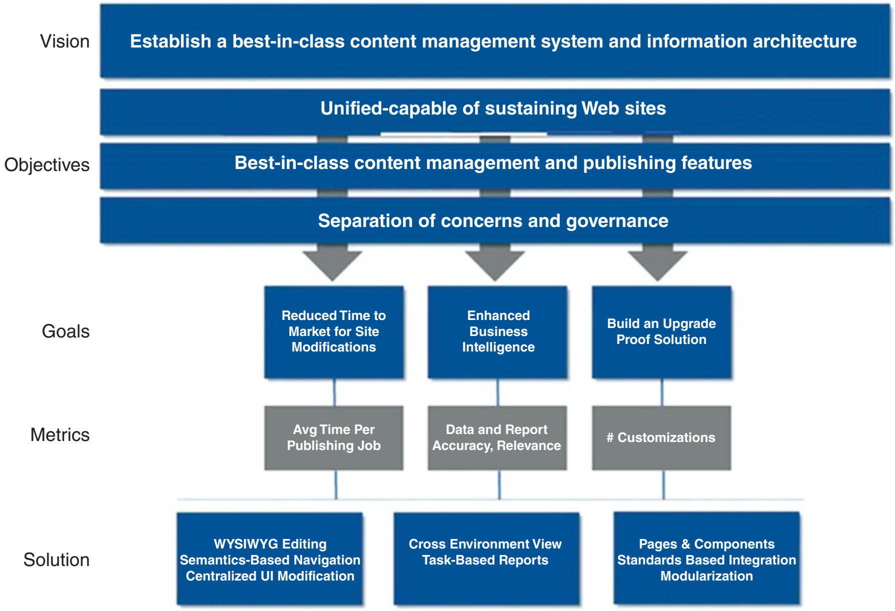

Vision
Establish a best-in-class content management system and information architecture
Objectives
Unified-capable of sustaining Web sites
Best-in-class content management and publishing features
Separation of concerns and governance
Goals
Reduced Time to Market for Site Modifications
Enhanced Business Intelligence
Build an Upgrade Proof Solution
Metrics
Avg Time Per Publishing Job
Data and Report Accuracy, Relevance
# Customizations
Solution
WYSIWYG Editing
Semantics-Based Navigation
Centralized UI Modification
Cross Environment View
Task-Based Reports
Pages & Components
Standards Based Integration
Modularization

Figure 4.3 Program vision mapping to CMS solution components

effectiveness. This is part of the solution definition phase we saw in CMS implementation approach. Figure 4.3 shows a sample mapping of program vision to CMS solution components and corresponding metrics.

In Figure 4.3 we can see the mapping of program goals to metrics and CMS solution components. CMS solution components such as author-friendly editors may improve productivity and reduce time to market. Leveraging built-in page CMS solution components and adopting standards-based integration makes the solution extensible for future enhancements.

In this way, we need to identify the key business goals and objectives and identify/build solution components for realizing them. If the CMS does not have any native solution component to fulfill a business goal, then we need to fill that gap through custom component development, or to identify open-source alternative. This mapping (along with tracking metrics) provides a framework for business to quantify and track the value of CMS.

4.4 CMS SOLUTION DESIGN PRINCIPLES

A CMS architect must define the key solution design principles during architecture and design phase. Solution design principles should be in line with industry best practices and align with strategic goals of the organization. These solution principles act as guidelines for solution leads and developers. Primary CMS solution design principles are listed as follows.

Layer-wise separation of concerns

In a CMS solution we need to identify the layers for handling a distinct responsibility. We have discussed this principle while discussing N-tier and N-layer architecture pattern. Typically in a content scenario, we can identify the following layers:

- Core content layer: This layer takes care of creation of content through templates and of its publishing. This includes CMS solution components such as content templates, content categories, tagging components, content versioning, deployment, workflow, users, roles, and access management.
- Presentation layer: Content presentation layer includes the rendition of content for a given presentation platform. Core content can be reused across multiple presentation platforms through XSL transformation or through presentation templates. CMS usually provides lightweight portal presentation engine for this layer.
- Content services layer: In this layer, key content functions (such as content authoring, content updating, and content search) are exposed as modular services.

Additionally there could be other layers, such as content security layer and content transformation layer, based on solution requirements.

Flexible template design

Authoring and presentation templates should be made flexible enough to support diverse data structures. Flexibility in content definition and publishing is to have the notion of “flex” fields in templates and layouts. Flex fields enhance reusability of the template for authoring a wide variety of content structures.

Every template or component definition should be based on the full spectrum of publishing options for content (in terms of the content type such as image, video, PDF, rich HTML), and every page template definition should account for variations in layout through flexible containers. This helps the author choose the template and layout to be used within a defined set of guidelines.

The built-in templates provided by the CMS product should be leveraged and extended to suit the requirements. In addition to supplementing standard page templates with “flex” templates from a page layout perspective, we should also design the native data capture templates and components keeping in mind flexibility of content entry fields. All potential options for content types that could be entered in a template need to be considered, and the variations should be made optional. There should also be variations in the page templates created to support the various resolutions, page layout, and output channels.

Since it is possible to control the type of components that could be dropped on any given page, the solution should be able to define the allowed configurations between the various standard, custom, and hybrid content/applicative page templates.

Functionality fitment and solution component mapping

This solution principle aims to fully cover the CMS solution from the business requirements standpoint. We need to focus on capturing the complete set of business goals and requirements to implement required features and functionality. We also need to identify the nonfunctional requirements (such as performance, security, scalability, availability, etc.) and map these requirements to solution components. Requirements could include localization, multi-site deployment, implementing complex business processes, and the like.

While mapping business requirements to CMS solution components, we can classify the "Functionality fitment" into these categories: "Fully compliant" (wherein the CMS components satisfies the requirement out of box or through configuration), "Partially compliant" (wherein the CMS components need to be customized or extended to implement the requirement), and "Not compliant" (no CMS component exists to match the requirement). We need to fill the gaps for partially compliant and not compliant requirements through open-source alternatives or custom components.

Operations self-service

Maintainability is one of the key solution considerations. Solutions fail as they cease to be maintainable, no matter how functional they are. This principle translates to following requirements:

- Enabling or leveraging self-service features for CMS administrators and business stakeholders. Self-service features can be enabled through intuitive administration interface that has minimal dependency on IT support teams.
- Ensuring minimal (preferably zero) customization to the core components.
- Separating customization from actual development that is allowed within the confines of the solution. Most CMS platforms provide extensions (or plugin) framework that can be used to extend the functionality. We need to use this for developing customizations.
- Following best practices while carrying out customizations ensuring a smoother upgrades and migration.
- Introducing operations considerations for every functional requirement.

Flexible integration

CMS solution should offer robust capabilities for integration with external systems. CMS should be able to expose and consume its core functionality through services and APIs in open standards. CMS solution should expose core content functions as REST-based or SOAP-based Web services with core content in XML or JSON format. If the CMS is JCR compliant, then clients can also use JCR APIs to perform content operations. We will be discussing integrations and data formats in more detail in Chapter 7.

The second category in integrations is CMS solution integration with external systems such as metadata systems, translation systems, document management systems, file repositories, commerce systems, etc. CMS should provide built-in connectors and integration components to connect to all required external systems. CMS should also provide an extensible plugin/connector architecture that can be used to build custom connectors.

Robust scalability and availability

A scalable application maintains optimal response times for the application even with increased user and content load. A scalable application optimally handles increasing amount of content overtime and process modifications. A well-thought-out deployment model, information architecture, development, and testing best practices are key to achieving scalability. Hardware sizing, clustered deployment, intelligent load balancers, and horizontal and vertical scaling are some of the best practices that can be done at infrastructure end for robust scalability. Usually authoring instance and publishing instance will be sized based on performance and load requirements.

High availability can be achieved through clustered setup, disaster eecovery (DR) setup, and removal of single point of failure (SPOF). Alternative options such as cloud hosting can also be considered for high scalability and availability.

Reliability

This includes process, data, and application reliability and is determined largely by solution architecture and design elements. This consideration is very important in a complex content management system where data is potentially derived from multiple sources.

Content security

Includes role-based access control, application, and data security. Core tenets of content security are authentication, authorization, fine-grained access controls to content, robust permission models, single sign-on (SSO), content auditing, and the like. Content performance is discussed in more detail in Chapter 11.

Performance

The solution should be designed to satisfy specific performance SLAs. The CMS and solution should support content caching (for caching content fragments), file caching, node caching (for caching JCR node), page caching (for caching entire page), asset caching (for caching digital assets), and other caching techniques for optimal performance. For higher performance requirements, content delivery networks (CDN) could also be leveraged. Content performance is discussed in more detail in Chapter 12.

Extensible and future proof

CMS solution should design forward-looking features into solution components. For instance, content-authoring templates should be designed to author content for multiple channels to cater to multiple devices and multiple languages in the future. Similarly, custom components should be designed so that they can be easily extended or configured for future needs. We can leverage the CMS product provided an extension framework (usually consisting of interface and abstract class based framework) to extend the core CMS features. On the infrastructure side, the hardware should be designed to handle potential growth.

Standards-based solution

The solution must use open standards so that it can be easily migrated or extended in future. Open content standards (discussed in Chapter 7) and industry integration standards should be adopted in the solution design.

So far we have taken a look at various architecture patterns and key CMS solution design principles. We will now focus our attention on the design of CMS solution components.

4.5 DESIGN OF CMS SOLUTION COMPONENTS

We have so far seen how the business requirements get mapped to CMS solution components during the CMS definition phase. As a part of this process, we also identify the gaps in existing CMS solution components and custom implementations needed to fill the gaps. The next phase of the CMS implementation approach is the development phase. In the development phase we design all the components needed to implement the solution. Diagrams depicting solution components are sometimes referred to as logical architecture or solution architecture depicting the CMS solution components.

Recall the core CMS components as depicted in Figure 3-2, which provides the solution view of the content ecosystem. Some of the core CMS components need to be elaborated on, and hence we have dedicated separate chapters for these discussions: templates, preview, and workflows are discussed in Chapter 5; tagging and metadata are discussed in Chapter 6; content security in Chapter 11; content services and content integration in Chapter 7; and digital asset management in Chapter 8. In this section we discuss the key components such as multi-site management design, localization design, URL design, and collaboration design of CMS.

Multi-Site Management Design

Multiple sites are required for globally deployed Web sites to serve global audiences spread across wide and disparate geographies. In some cases of multi-site architecture, the authoring/publishing instances are located in different geographies for performance optimization purposes; in other cases they are deployed centrally. CMS should support role-based access to site deployment process and should provide remote content deployment capabilities. The main use cases of multi-site CMS are:

- An enterprise has Web sites specific for each country. The corporate site owns the content such as product releases, FAQ, company policies, etc. So when there is any change to the corporate-owned content, the changes should be propagated to all country-specific sites.
- An enterprise owns Web sites in multiple languages, and we need to manage this multilingual content. Some of the common multi-site management activities in this regard are:
  - It is required to maintain the corporate branding elements such as branding design, logo, and main look and feel to be consistent across all Web sites. This requires creation and management of common content elements.
  - The content hierarchy of various geo-specific or country-specific sites would vary based on the specific country or geography.
  - Handling of content fallback rules for translated content should be available.
  - Separate hardware for each region to allow site-specific scaling for the respective site and for site-specific availability.
  - The country-specific site can use a single language or various languages (such as India regions supporting both Hindi and English). This depends on variation of languages used by the target audience.

Multi-site management framework The main elements of a multi-site management framework are depicted in Figure 4.4.

The multi-site management framework has four main components:

Multi-site content: As part of multi-site management, we need to have authoring templates to support content structure required for all languages and geographies. The content may need to be localized (through resource bundles and translation process) needed for all sites. The workflows should handle the language and site-specific business rules and routing logic. Content translation is one of the key steps in the publishing workflow. The last step in the publishing workflow transforms the localized content to the desired presentation format.

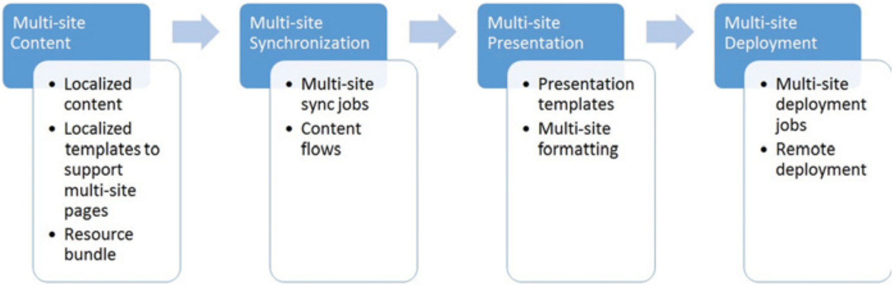


Figure 4.4 Multi-site management framework

Multi-site synchronization: Synchronization jobs would copy the content from master site to all remote sites. Content workflows can also be used for synchronizing content across all sites.

Multi-site presentation: The presentation transformation engine (or presentation templates) should support localized presentation and any required variations (such as handling right-to-left formatting). Any site-specific formatting should be supported.

Multi-site deployment: After the content is transformed, the content is deployed to multiple sites. This also includes remote deployment to geo-specific sites. The presentation platform should render localized content (it can use using locale detection techniques and such).

Multi-site design In this section we look at intricate details and various models for multi-site design.

Usually multi-site management happens through asset/content replication and through asset/content synchronization across primary site and remote site.

Primary/master site owns the core content and assets such as core content modules, plugins, libraries, master style sheet definitions, core frameworks, analytics libraries, common content templates, core workflows, integration modules, site taxonomy, master content, and common content and assets. Essentially primary/master site owns all the common elements (such as UI branding assets, common library components, metadata, and taxonomy). Hence, master site forms the centralized content and assets repository for replication.

Remote site contains site/country/geography/language-specific files, style sheets, page layouts, geo-specific content templates, site-specific assets, page layouts, navigation elements, site structure, and any site-specific workflows. Remote sites can either own site-specific content and assets or they can get it from primary site through synchronization. The generally followed best practice is to encode the multilingual site pages with UTF-8 encoding so that they can support multi-byte character set.

During publishing, CMS supports multi-site publishing wherein the site-specific localized content will be published across various remote sites through multi-site publishing workflows. While the core assets (such as framework libraries, core CMS components, analytics frameworks, branding assets, etc.) will be shared across primary and remote sites, site-specific content will be copied during publishing to the remote site. Sometimes some of the core components may need to be modified for each of the remote sites to satisfy business requirements. For instance, a page template may need slight modification for some languages. In such cases the core assets will be replicated across each of the remote sites and then they are customized as per site-specific requirements. Country- and geo-specific sites also subscribe to the feeds for the content updates from the primary site. For instance, since company policy content is owned by the primary site, all country-specific sites will be notified about the policy content updates so that they can update the content accordingly. The multi-site publishing workflows are strictly controlled by role-based access permissions.

The content administrator creates locale-specific sites, which are modeled mainly on the primary site. New locale-specific sites are created through these steps:

- Administrator copies the primary master site along with all templates, libraries, workflows, and components. Content templates are normally "owned" by the master site to ensure easier governance.
- Any required variations in the site and page structure will be done for each of the locale sites.
- Administrator configures the necessary security roles and permissions for each of the locale sites.
- Administrator sets up the content synchronization process between primary site and the locale site. This will be used to update the master templates and other shared components across primary site and locale-specific sites.
- All required base content would be translated to locale languages to suit the site-specific needs.

There are mainly two models for handling multi-site authoring and publishing:

- **Centralized authoring and deployment**: In this option, all content activities (such as template design, authoring, workflow management, localization, publishing, security management) are managed centrally at the primary site. There are two variations in the deployment. The publishing workflow can deploy to the centralized deployment location (such as enterprise Web server), and end-users (including users from various geographies) will access from this centralized location. In the second variation, the publishing workflow will do a remote deployment to site-specific servers distributed across geographies. In this scenario end-users will access from site-specific sites.
- **Distributed authoring and deployment**: In this option, each of the sites has their own version of authoring and publishing instances. These geospecific sites inherit common structure and code base from the primary master site. Authors can author site-specific content, and the deployment workflow will deploy only to their specific sites.

When to use centralized authoring vs. distributed authoring? Factors such as governance processes, geo-location of content authors, content load, and performance need to be considered while deciding centralized vs. distributed authoring. If the organization needs to have a centralized control over all authoring and publishing governance processes and if all content authors and editors are co-located, then it is better to go for centralized authoring.

On the other hand, if we have distributed authors (such as authors belonging to different geographies and locales) across various geographies and if there are performance/security issues to use centralized authoring instance, then we need to consider distributed authoring. If the site-specific variations (layout/structure, content, design, libraries, components) are extensive, then we need to consider distributed authoring.

The next section discusses the distributed authoring and deployment in greater detail.

Distributed Authoring and Deployment In some scenarios the authoring environment is distributed across various geographies, especially for supporting multilingual content editing. In such scenarios each geo-site can have its own authoring and publishing environments.

The main design aspects of multi-site authoring setup are as follows:

- Primary site "owns" the key branding elements, visual style guides, page layouts, core content (such as company policies, legal documents), core CMS components, and main processes. If the geo- or locale-specific sites have significant variations in these aspects, they will be reviewed and the components extended or customized accordingly. In a majority of cases, these primary components are shared and reused across all sites.
- The main content hierarchy is defined at the centralized primary site. This includes the site taxonomy, page hierarchy, page flow, and others. All geo-specific authoring environments inherit these elements from the primary site and reuse them for authoring.
- The metadata and content categories are also defined at the primary site (including site-specific metadata). It will be shared with various remote sites.
- The authoring and presentation templates will also be defined at the primary site. Similarly, all common components such as header/footer elements, navigation elements, custom components, branding elements, and style sheet definitions will be developed at primary site level and all the geo-specific authoring sites will reuse/inherit this; they will use it for authoring content in geo-specific locale. Unless there are compelling reasons, the structure of these templates will not be changed.
- If there are any variations in terms of page structure or custom components, then content architects will review it and the master components will be extended to suit the geo-specific variations. Such variations are also handled by customizing workflows by adding geo-specific business rules.
- Security roles will be defined for geo-specific authors to ensure appropriate role-based access. All geo-sites will connect to organization-wide user repository for uniform definition of user roles; for performance optimization, the user roles and other attribute details may be provisioned to the geo-specific sites.
- Each geo-site will reuse the publishing workflows for publishing to its local publishing folder. We will discuss the publishing and deployment strategy in the next section.

Note: In case of product catalog content (in e-commerce scenario), it is better to maintain a separate product catalog for each country to minimize performance impact due to huge catalog size.

Multi-site deployment strategy This section discusses main strategies commonly used for multi-site deployment. The methodologies also consider multiple languages at each of the sites.

Live Copy Approach Live copy approach allows for multiple copies of the master site to be created and configured for each locale that inherit the same code base, functionality, and features of the master site.

Salient features of the live copy approach:

- Live copy approach allows inheritance of the approved master site for a particular locale. The locale-specific configurations are done through localization configurations.
- Multiple sites share the same code base, increasing efficiency and providing greater control on the functionalities made available on the site.
- This is one of the quickest methods to roll out the sites across multiple locales.

This approach can be used where there is no locale-specific customization (in terms of layout, business rules, workflow logic, etc.) The locale site is the exact replica of the master site; the only difference is translated content.

Partial Localization Approach There are many instances where we need to adopt the features available on the master site but with some customization specific to the locale where the site is to be rolled out. In such scenarios, the following approach can be used. In this model the customizations at the locale site happen simultaneously while the live copy is done from the master site.

Salient features of partial localization:

- Locale site can be released for authoring early.
- Customizations at locale site can be done in parallel.

We can adopt this approach if there are many locale-specific variations that need to be handled at the locale site.

Multi-site synchronization The assets from the primary master site are synchronized with the locale-specific sites through workflow and synchronization jobs. The main candidates for synchronization are:

- Content updates (addition, updates, deletion) when the core content (policy document, key design elements, etc.) are updated
- Library updates when the CMS libraries are updated
- Component updates when core CMS components are updated.

Based on solution needs, other components can be marked for synchronization. The synchronization workflow notifies all the locale-specific sites when any of the above components gets updated. In case of a content update, it will also trigger the translation workflow to create the localized versions of the updated content.

Synchronization jobs normally perform batch synchronization between the primary site and the remote site.

We will discuss multi-site deployment in a case study in content case study chapter hosted in book support web site.

Content Folder Design

Once the content is authored, it is saved to a particular folder location. Content folders should depict the logical grouping of content. The folders could be modeled based on categories such as site sections, content categories, business structure, languages, geos, and the like. All CMS platforms provide flexible content folder structure wherein CMS administrator can create solution-specific content folders.

Sometimes content metadata values play a role in deciding storage folders. For instance, when the author tags “en_US” as the language for content chunk, that content chunk is stored in the “en_US” folder. Similarly, the author can select any particular content structure or content type while authoring and metadata decides on the content type–specific folder location.

Content URL Design

Designing an optimal URL structure for the site content is the joint responsibility of a CMS administrator and a content strategist. The content URL should reflect the content hierarchy and the URL should be structured and flexible. Normally the content URL is closely modeled on the folder structure or content hierarchy in the content repository. The URL pattern should be flexible and scalable to include the language, geo, and other future needs. Normally the content strategist will lead the content URL design leveraging the CMS capabilities.

There are mainly three URL structures: tree-type/hierarchical URL, flat URL, and dynamic URL.

- In **tree type/hierarchical URL structure**, the URL structure would model the current content in a content repository. Each content subfolder becomes a part of the URL path. The URL structure is consistent with underlying content folders, site taxonomy, and other navigation elements (such as left navigation, breadcrumb). This is the most commonly used URL structure as it provides flexibility to add additional content items, folders, and subfolders. For instance, the URL http://server/us/en/electronics/tv/led is modeled on the underlying product content hierarchy or content category model. Content hierarchy or content category is normally modeled in the taxonomy.
- In the **flat structure URL structure**, the URL will be linear with minimal hierarchy. This would be appropriate if there is no logical content hierarchy in the underlying content repository (like list of information pages or descriptive URL for news items). For instance, the URL http://server/us/en/electronics-tv-led has a flat structure. In this case the product content hierarchy information is

flattened out and has minimal hierarchy information (though we have URL paths to depict language and geography).

- A **dynamic URL structure** is generated at run time with no fixed pattern. Dynamic URLs often contain encoded state information, navigation information, session information, and other similar information. For instance, the URL http://server/us/en/eiewqsoels-widl is a dynamic URL that has no fixed pattern. Usually dynamic URLs are system-generated URLs. It is always recommended to map it to a friendlier format. Another variant of the dynamic URL is http://server/us/en/products?product_id = 1234&product_category = electronics

The URL structure also impacts the site SEO (search engine optimization). We are going to look at SEO in greater details in Chapter 10.

Table 4.3 shows the comparison of flat and hierarchical URL models:

While the hierarchical URL structure is preferred in a vast majority of cases, we need to look carefully at each scenario to evaluate the best URL strategy on a case-by-case basis. We can use the points given in Table 4.3 to decide on the optimal URL model for a given scenario.

In some cases we can have a "hybrid" URL structure that can contain the mix of both hierarchical and flat structures. This is used when there are too many folders and subfolders, which may impact the URL usability. In such cases we will depict the first few key folders and then flatten the URL structure:

Table 4.3 Flat vs. hierarchical URL model

	Hierarchical URL Structure	Flat URL Structure
URL pattern	The URL depicts the content hierarchy and is consistent with site structure and user navigation.	The URL has a flat structure without any hierarchies.
Usage scenario	We can use this when the underlying content is structured and hierarchical. A classic example is that of product content that contains product category and subcategory hierarchy.	We can use it to model a static content structure, which has minimal content hierarchies. Examples of such structure are campaign pages, microsite pages, news content, blogs, how-to solution articles, quick links, etc.
SEO impact	Hierarchical URL helps search engine indexing through categorization and structure.	Flat-structure URL provides the main keywords as part of URL to search engines.
Usability	User can easily remember and bookmark a structured and hierarchical URL. However, if the URL is too long, then it may impact usability.	Usability is more for short URLs and for static content.

http://server/us/en/category/subcategory1/subsubcat1-subsubsubcat1-product-name.html

While selecting a URL structure, the content strategist should take note of future content needs and design a flexible URL structure to support to those needs. For instance, if there is a need to serve the content to various geographies and languages, then the URL structure should include the language and geo-variation patterns so that the URL structure can be extended for language/geo-variations in the future.

Best practices in URL design Irrespective of the URL strategy we choose, there are some best practices from the SEO standpoint:

- Try to use the relevant keywords in the URL. Include product title and/or content title in the URL.
- As much as possible, use the exact product name or descriptive product name in the URL.
- Use hyphen ("-") as a separator in a flat URL structure.
- Ensure that key pages are reachable within maximum three user clicks.
- Ensure consistency between site structure and URL structure in a hierarchical URL pattern.
- URL structure should accommodate future road map plans (such as expansion to new geographies, new languages, etc.).
- Dynamic and cryptic URLs must be avoided and should be mapped to user-friendly formats.
- Convert dynamic URLs to a hierarchical structure wherever possible. For instance, URL http://server/product?id=234 can be converted into http://server/product/1234
- Redundant and unnecessary categories should be eliminated from both URLs and site structures.
- Avoid lengthy URLs, as this would pose usability challenges. A maximum URL length range should be 40 to 60 characters.

Localization Design

Creation and management of localized content is one of the main use cases for a content management system. Globally distributed sites should cater to various geographies and should render content in localized languages. CMS allows localization mainly through these features: built-in language packs and plugins (for localizing the user interfaces), localization workflows (for content translation), and multi-site management features (which we discussed earlier). Additionally, CMS should allow for UTF-8 content to support additional languages in the future.

Localization requirements

We need to understand the needs of all geographies and locales from a content perspective. Given below are some of the main points in this regard:

- The main requirement is related to content translation. We need to understand the complete content translation requirements (such as page sections that need translation, legal content that needs translation, header/footer content needing translation, look-up list needing translation, support content, product content, etc.).
- Localization workflow: We also need to capture all geo- and language-specific variations of workflow steps. This includes any locale-specific business rules, extra or different process steps to comply with local regulations, and so forth. For instance, localized content needs to be reviewed by the person proficient in that particular language, and hence the skill-set-based workflow routing logic needs to take care of this.
- Localized metadata: Normally metadata attributes such as language, geography, region, locale, and others are used for tagging and locating content across various geographies and languages. In addition to these commonly used attributes, we also need to understand if we need extra metadata information or tags to identify the locale-specific content.
- Localization NFR requirements: We need to understand the nonfunctional requirements such as performance SLAs, availability SLAs, and scalability SLAs for each language/geography.
- Any language or geographically specific visual style guide or font specifications need to be captured.
- Any other special requirements needed for the target geography and language (such as geo-specific site accessibility standards) should be captured.
- Any site or page layout variations (such as right-to-left layout) required for the geography and language need to be captured.
- All supported character sets for various locales need to be documented.

Note: For distributed geo-specific sites, we have covered the deployment design as part of multi-site management topic.

CMS capabilities needed for localization

Localized content is required to cater to audience from various geographies. CMS should be able to support the following features to support localization:

- Ability to support various languages out of the box. This includes ability to localize the page elements and to support for adding language resource bundles, language packs, etc.
- Ability to add new language

- Support for localized metadata and CSS/asset variation based on languages
- Support for multi-byte character set and UTF-8 character set
- Support for translation workflow and easy integration with a translation management system (TMS)
- Support for format variations (such as geo-specific variations in date, time, calendar, currency, phone number format, etc.), site-specific encodings, and other application-specific localization requirements.
- Localization support for Web content and assets
- Help and supporting documentation in local languages.

Content localization

As we have seen in an earlier discussion, content translation is an important step in localization normally implemented through workflows. The general steps for achieving localization include the following:

- Integrate CMS with a translation management system (TMS). This is usually done through services or API-based integration. More on this step in Chapter 7.
- All other CMS integrations need to be tested for locale-specific scenarios. Service invocation calls need to be validated with localized string values, multibyte characters, and different formats with all kinds of character encoding to ensure that integration works for all locales.
- Site taxonomy may vary based on geographies and languages. The taxonomy structure and URL structure should be flexible to handle this variation.
- Metadata and tagging interfaces and structure should be able to accommodate any variations specific to the geography or the language.
- Create and maintain the master copy of content. The English version of content is normally used as base (or master) version for further translations. However, we also need to create entirely new localized content copies on the need basis.
- Some of the localized pages need a different layout and structure, so we need to create appropriate page templates for them. It should also support any other special requirements, such as right-to-left compatible layouts.
- We need locale-specific graphics if they have embedded text information.
- We need to create locale-specific product data along with its image for serving various geographies.
- Develop the localization workflows that include all necessary review, approval steps, and any applicable business rules. The workflow is configured to invoke the translations services exposed by the TMS. Key features of localization workflow are given below:
  - Initiate the localization workflow for master content copy, which requires localization. Usually both master copy and the localized copy will be tagged with the same metadata with the exception of language and geo-tags.

- Workflows should be able to have locale-specific variations
    - Locale-specific business rules that comply with local regulations
    - Support for process changes due to local legal regulations
    - Differences in financial or cultural aspects
- Once localization workflow is completed, author and approver preview the localized content for correctness.
- Localized testing: All features (such as templates, workflow, integrations, and deployment) should be tested for locale-specific scenarios with localized test data. Normally localization testing is carried out with the help of native-language testers.
- Multi-site deployment: Content from the master site will be deployed to all remote servers. To support remote server deployment, the publishing workflow and deployment jobs should be able to support multi-site remote deployments.

The above steps explain the localization of core content. Content platform also has other internal features (such as CMS administration interface, template builder tool, header footer elements, etc.) that need to be localized. For this we mainly use the resource bundles (configuration file storing the key value pairs of labels for all languages) and language packs. Most CMS platforms provide extensive support for resource bundles and language packs. We may need to extend that feature for adding any new languages.

Collaboration Design

Knowledge sharing and collaboration is a prominent use case for content platforms. Hence, most of the CMS platforms provide native collaborative features and support social and collaboration plugins. Most of the collaborative features are content heavy (more user-generated content), and hence it becomes a natural use case for CMS platform. In this section we discuss key design aspects of collaboration features.

What follows is a list of most common collaboration features needed to build a collaboration platform using CMS:

- Social platform integration and plugins: CMS platform need to be integrated with external social platforms such as Facebook, LinkedIn, or Twitter. In many cases CMS provides social plugins to implement features such as content-sharing plugins, plugins for providing features, such as Facebook's "like," social search, posting and reading social comments, etc. Most commonly used integration mechanism is OAuth authentication-based and API-based integration. We will discuss certain integration methods in Chapter 7.
- Knowledge management systems: CMS platforms can be readily used to build knowledge management platforms. CMS templates can be used to build knowledge base articles, and workflows can be used to manage their publishing and localization. Content tags and metadata coupled with search can be used to

discover the relevant knowledge assets. We discuss an elaborate case study for building a knowledge management system at the end of this chapter.

- Blogs: This feature enables users to reflect their opinions and share their views or perspectives on various things. WordPress is specialized in blogging, while other CMS implements blogging through specialized page templates, rich text editors, notification, metadata, and supporting user-generated content.
- Wiki: This feature captures comprehensive information about a topic that can be edited collaboratively. Existing CMS components such as templates, workflows, permission model, and tags can be used for this feature.
- Community: This feature allows end-users to form groups based on their common interests and purposes. Community members exchange ideas and share documents and content.
- Forums: Provides a collaborative platform for group discussions related to a topic.
- Community calendar: The calendar tool helps community members organize and notify about community events. This is normally implemented as a widget and supports other features such as alerts, reminders, etc.
- Chat: Various forms of chats such as audio chat, video chat, and Web conferencing allow users to exchange instant messages and perform activities such as co-browsing, co-shopping, etc. This is another popular collaboration tool.
- Feeds: This feature allows users to expose content as feed (such as RSS or ATOM feeds) to the external world.
- File sharing: This collaborative feature allows users to share files and documents.
- People finder: Users can search people based on user attributes, skill set, interests, etc.
- Message boards: Similar to communities, this feature supports user conversations.

Besides these common features, a collaborative platform also includes other features such as polls, social tags, e-mail, message boards, review and rating, feedback, social sharing, ask and answer, dashboards, and others. Most of the CMS platforms provide native support or plugin support for implementing collaborative features. For instance, the Drupal Commons module can be used for collaboration in Drupal CMS, while Liferay Social Office can be used in the Liferay portal.

4.6 CMS OPERATIONS MANAGEMENT

This section takes an operations view of CMS. We also touch base here on release management, code promotion, and continuous content editing processes. Some of the CMS maintenance activities such as CMS backup, monitoring, patching, and upgrade are discussed as well.

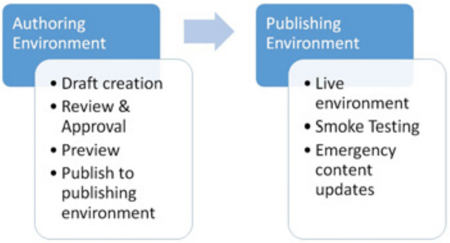


Figure 4.5 Content Publishing Process

Release Management Process

In this section we look at various steps involved in publishing content to a production environment.

Reviewed content and assets are copied from an authoring server to a publishing server as part of a publishing workflow. The process is explained in Figure 4.5.

An improvement over the aforementioned deployment model is to have a staging environment where approved content will be "staged" for review by business stakeholders and the marketing team. The staging pre-production environment can act as a UAT test environment and can be used to test the system performance. The revised deployment model is shown in Figure 4.6.

Content promotion process

As part of release management, content moves from one environment to another while undergoing review and testing.

Figure 4.7 shows the CMS environment hierarchy consisting of content-authoring and -publishing environments.

The environment hierarchy in Figure 4.7 depicts a robust process for code and content promotion while at the same time keeping the number of environments to a minimum.

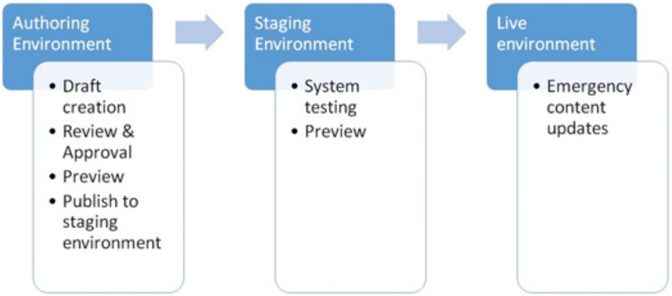


Figure 4.6 Content Staging Scenario

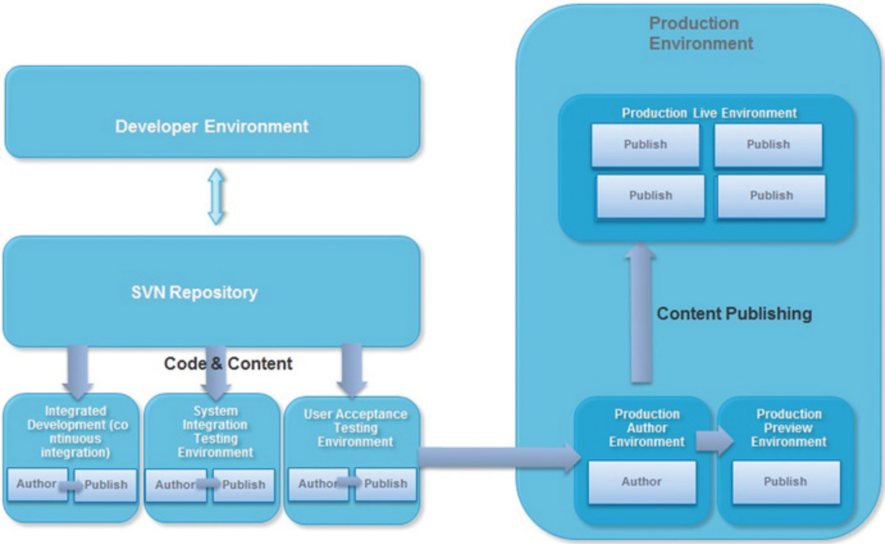

Developer Environment
SVN Repository
Code & Content
Integrated Development (co ntinuous integration)
Author
Publish
System Integration Testing Environment
Author
Publish
User Acceptance Testing Environment
Author
Publish
Production Environment
Production Live Environment
Publish
Publish
Publish
Publish
Content Publishing
Production Author Environment
Author
Production Preview Environment
Publish

Figure 4.7 CMS Environment Hierarchy and Promotion Process

The salient features of the code and content deployment process are:

- In the development environment, developed artifacts are checked into a source control system such as SVN. Artifacts in this phase include templates, Omnichannel content, CMS components, workflows, services, etc.
- Code from source control can be deployed to various environments. Code and content deployment to environments are automated (using continuous integration or configuration management tools). We will discuss this in the next section.
- The code promotion is controlled through development, system integration testing environment, user acceptance testing environment, and production environments for thorough validation.
- CMS-provided synchronization jobs are used to replicate the code and content from UAT into a production environment once all validation is complete.
- Production author environment can deploy content for preview and live-publish opportunities.

Continuous Content-Editing Process (for emergency updates)

Once content is promoted to the production environment, generally it is first published to a preview environment for verification. Post-verification and smoke testing, it is pushed to the production live environment (which is accessible to Internet users). In this section we discuss the continuous content-editing process for incorporating emergency content updates in the production environment.

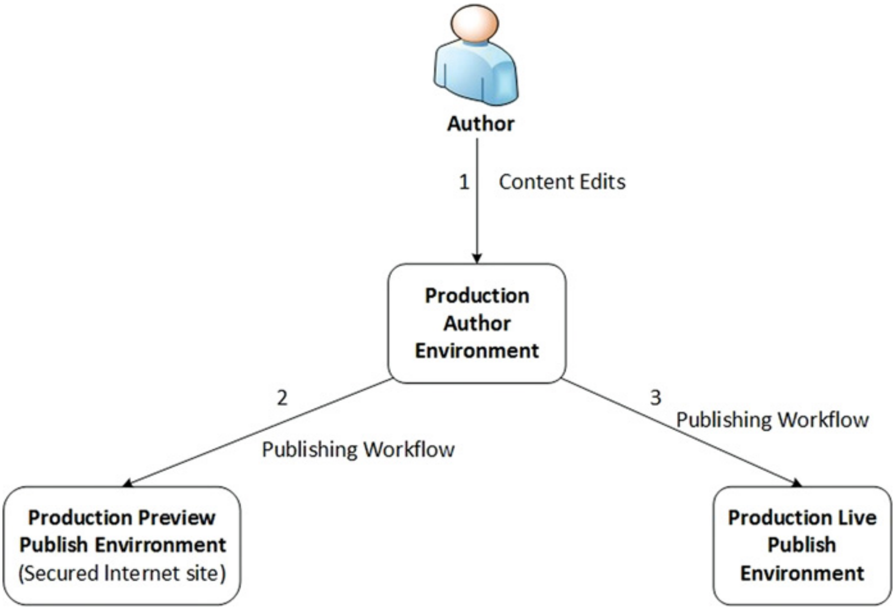

Author
1 Content Edits
Production Author Environment
2
Publishing Workflow
3
Publishing Workflow
Production Preview Publish Envirronment (Secured Internet site)
Production Live Publish Environment

Figure 4.8 Continuous Content-Editing Process

Figure 4.8 shows the process for continuous content edits on the site:

In a regular approach, the content change has to go through multiple cycles starting from development and through to production. However, in case of emergency content updates, we need to edit production content directly. This should be treated as an exception, and it brings agility in the process.

The steps to be followed are:

- Access-restricted control is given to authors for the production authoring environment. Security measures such as role-based access and author machine IP checks are done before access is provided.
- There are three environments in the production group:
  a. Production Author
  b. Production Preview Publish
  c. Production Live Publish
- Content authors modify content in the Production Author environment directly. Once this is completed, the authors initiate the workflow to promote content to the Production Preview Publish environment. The workflow also supports a versioning feature to facilitate content rollback when required.
- The Production Preview Publish environment allows the sites to be accessed over the Internet but protected behind the login credentials that prevent automatic crawling of the site by search engine crawlers. The preview environment can be used by an external creative agency or remote validation teams for validating content.

- The Production Preview Publish environment–based sites can be reviewed for content correctness over multiple browsers as well as handheld devices. Any kind of additional approvals required for content (such as regulatory, compliance approvals) can also be obtained from a site based on this environment.
- Once edited content is reviewed on the preview site and found to be appropriate, the content reviewers/publishers can approve the content through the workflow and then move it to the Production Live Publish environment.
- Approved content is also “back-filled” to the centralized source control system as part of the reverse synchronization process. Usually production updates will go into production release branches.

CMS Maintenance

Monitoring and maintaining CMS are essential parts of CMS operations governance. It ensures the quality and turnaround time for iterative releases. In this section we take a look at some of the key CMS maintenance activities.

CMS Backup

CMS mainly use content repository and database for storing code and configurations. Content repository and database data must be backed up regularly. CMS administrators can create jobs to make automatic periodic backups. The main design considerations for CMS backup jobs are as follows:

- The CMS backup jobs should be capable of making a complete backup (of entire content from content repository, data from database, assets from DAM, related documents and configurations from document systems) and partial backup (of only updated/changed content since last backup).
- The backup jobs should be scheduled to run during minimal traffic hours so that they do not add any performance overhead to the CMS. We can monitor the site traffic patterns and understand the minimal traffic period and we can then configure the backup jobs to run in those time periods.
- Based on content volume and content update frequency, we can configure to run the complete backup jobs and partial backup jobs. Usually, complete backup jobs run once a week and partial backup jobs run daily. This again can be fine-tuned based on business criticality of data and content update frequency. Based on these, we may also need to increase the frequency of partial backup jobs.
- RPO and RTO: Recovery point objective (RPO) specifies maximum allowable data loss in a time period. For instance, if RPO is two hours, it means that the business can tolerate data loss for a maximum of two hours. RPO plays a vital role in determining the frequency of backup jobs; the smaller the RPO, the higher the frequency of backups. Recovery time objective (RTO) is the

maximum time period within which the system must be restored after a failure. RTO indicates the scope of data that needs to be backed up. The smaller the RTO, the wider the scope of backup data.

- The backup job should also back up the critical system files and configuration files used by CMS.

CMS platforms provide built-in backup and replication jobs that can be leveraged for this. For instance, we can use the "Backup and migrate" module in Drupal to back up the database, code, and files and to synchronize them across environments.

Monitoring

As part of CMS maintenance, we should set up real-time jobs that monitor the content servers and notify in case of any exceptions. Table 4.4 summarizes key systems and components that need to be monitored.

Patches and upgrades

In the CMS world, a "Patch" is a package of changed code. Sometimes patches are also termed as "module" or "plugin" in some scenarios. Patches are used for fixing a known defect, extending core functionality, addressing security vulnerability, or implementing an enhancement. CMS must be patched regularly based on the latest releases from a CMS vendor. The patches include, among others, bug fixes, security upgrades, performance enhancements, and feature enhancements. Patches and upgrades are tested on staging environments to verify the impact and, after thorough testing, are applied to production systems.

The following list presents some of the common best practices for patch management:

- Tracking updated components: When developers are releasing patches, they should clearly document the files changed and functionality impacted and provide the exact version number for source code.
- Backtracking plan: Before applying a patch, it must be tested in a pre-production environment to understand the impact. There should be a well-defined plan to backtrack a patch installation in case of any unexpected failures. To achieve this we need to completely back up the production code base in a file system or as a production release branch in a source control system.
- Impact communication: All stakeholders and parties should be communicated about the patching schedule beforehand, and we need to identify any potential impact (such as application downtime, non-availability of functionality, impact on performance, etc.)
- Minimizing downtime and maximizing application availability: We need to use patching strategies to minimize downtime and maximize the availability of the application. One of the effective patching strategies in this regard is "rolling patching," which ensures continuous availability of the application. In a rolling

Table 4.4 CMS monitoring activity

System	Monitored Activity	Comments
Authoring Server	CPU Usage	High CPU usage points to overloaded system. Scale up the hardware through appropriate sizing. Setup a clustered nodes of authoring server to distribute load
	Memory Usage	High memory usage points to an overloaded system. Scale up the hardware through appropriate sizing. Set up clustered nodes of the authoring server to distribute the load.
	Performance (Response times and load times for authoring systems)	Use appropriate sizing for the server hardware. Enable server-side caching and session replication for optimal response times.
	Disk Usage	Allocate more disk space if the maximum limit is reached.
Publishing Server	CPU Usage	Use appropriate sizing for the server hardware.
	Memory Usage	Use appropriate sizing for the server hardware.
	Page Performance (Page response times and load times)	Use Web performance optimization techniques such as page caching, asset caching, asynchronous request servicing, leverage CDN caching, and Web server caching.
	Disk Usage	Allocate more disk space if the maximum limit is reached.
Network	Bandwidth Usage	Increase network capacity based on requirements.
Logs	Request Log System Logs	Watch out for any server exceptions and security incidents and take appropriate action.
Database	Query Response Time	Use query cache and fine-tune the query. Increase the capacity of the database server if the response time is low.
Dependent systems	DAM Performance, Metadata Management System (MMS) Performance, Database Performance	Monitor the performance of all dependent systems and notify the system owners to take appropriate action.
Dependent services	Localization Services Performance, Content Validation Service, Feeds	Monitor performance of all dependent services/feeds and notify the system owners to take appropriate action.

patch process, patching happens one node at a time in a multi-node/clustered scenario. The node being patched is taken out of the cluster (so that it does not service user requests) while other nodes remain active. Once the patch is applied and tested, the node is added back to the cluster. This process is repeated for all cluster nodes. Some servers provide a "node synchronization" feature that synchronizes code/configuration changes from primary node to all cluster nodes. This feature can also be used for implementing a rolling patch.

4.7 REALIZING CONTENT STRATEGY WITH CMS

In Chapter 2 we have discussed various elements and phases of content strategy. CMS plays a key role in the content strategy execution phase for designing and creating content. Let us examine how we can leverage CMS for content strategy execution phase.

As we know, in the execution phase of the content strategy, we mainly focus on content design and creation. CMS provides built-in components to aid in the design and creation of content.

Content Design and Creation Using CMS

Table 4.5 maps various CMS components/artifacts that can be used to implement the content design and creation activities:

4.8 CMS REFERENCE ARCHITECTURES

CMS can be effectively used to build robust solutions for various business domains. We have discussed the utility of CMS in Chapter 3.

In this section we look at how we can build robust CMS-based solution for various business domains. We will examine reference architectures for a customer experience platform, knowledge management system, and digital marketing platform. The reference architecture serves as a guideline for CMS and content architects while designing solutions.

Customer Experience Platform (CXP) Reference Architecture

Customer experience management (CXM) is an emerging popular trend for Web applications. CXMs are also referred to as digital experience platform (DXP) and user experience platform (UXP).

CXM involves managing and optimizing the end-user experience at all touch points throughout user journey. CXM uses a suite of technologies and aims to provide compelling and engaging user experiences, process improvements, and business alignment. Generally, CXM technologies are lightweight in nature and are best suited for agile engagements.

Table 4.5 CMS components for design activity

Content Design Activity	CMS Solution Component and Comments
Content workflow design	Built-in workflow modeling tool (such as Kaleo designer in Liferay CMS) *Comments: Workflow modeling tools can be used for optimized modeling of content authoring, publishing, translation, and other processes.*
Content template design	Built-in authoring and presentation templates *Comments: Most CMS provide tools to design content-authoring and presentation templates. Built-in and custom templates can be designed to author site content. We will discuss this in more detail in Chapter 5.*
Content modeling	Custom templates *Comments: Once the required content models are identified, we can create custom templates in CMS based on those content models. Custom templates should structure all the elements of the content model and enforce any associated rules (such as element sequence, allowed values, etc.). Custom and built-in templates can be used for content creation.*
Content design	Built-in rich text editors, CMS support for custom styles, and JavaScript frameworks Content designers can also use different tools for designing wireframes and visual style guides.
Metadata design	Built-in CMS metadata support can be leveraged or CMS can be integrated with enterprise-wide metadata management system (MMS) *Comments: Authoring templates should provide fields for adding semantic metadata tags.*
Content security design	Built-in permission model and role-based access control, built-in access control list, native authentication handlers, and security plugins *Comments: Most CMS provide an administration console to assign roles to various resources. We can leverage this for the security design.*
Content copy decks	Author content using authoring templates *Comments: Copy decks can be used as references and job aids for content creation according to laid-out specifications.*
Asset design	Built-in support for assets or integration with DAM for asset design (such as asset tagging, asset resizing, asset workflows, etc.)
Omni-channel content design	Adaptive content modules of CMS *Comment: CMS support for adaptive content, responsive page layouts, and related modules can be leveraged for Omni-channel content design and content creation.*
Content distribution plan	CMS-provided services framework, multi-site CMS publishing features *Comment: Most CMS provide content services to expose content as SOAP-based or REST-based Web services. This feature can be leveraged to expose created content. CMS also provide multi-site publishing, which can be used for publishing content in various formats to different sites.*

Most product vendors offer a customer experience suite of products. Since it is a popular trend among B2C applications, we have detailed reference architecture of CXM using CMS.

CXM capabilities

Capabilities of a typical CXM platform are depicted in Figure 4.9:

Let us look at the salient features of a CXM platform; we will then leverage CMS for creating a CXP.

Seamless Omni-channel experience: A CXM platform should provide a seamless, responsive, and interactive experience for all access channels. The experience should be optimized for Web and mobile.

Rich user experience, delivered through interactive widgets, responsive forms, Omni-channel experience and personalized delivery

Real-time customer support, delivered through various channels such as real-time chat, phone, e-mail, Web conference, etc.

Business-friendly administration interface for managing workflows, search, personalization, and content management

Marketing and sales enablement: CXM boosts marketing efforts through highly targeted content delivery. Many CXM platforms would also provide multivariate testing and campaign management features.

Analytics-driven insights gathering: CXM uses Web analytics to obtain insights about customer behavior, and we can use to personalize the experience. The gathered data can be used for personalized recommendations and targeted content delivery.

Active user engagement: User engagement is the prime focus area of CXM. The platform provides responsive and interactive user experience through rich media assets, widgets, mashups, and responsive presentation components.

Social and collaboration: CXM integrates with social platforms and provides collaboration features to have meaningful conversations with customers. Collaboration features such as blog, wiki, communities, and forums may harness collective intelligence.

Optimized processes: Business and support processes would be optimized to provide differentiated experience. This includes process simplification, process automation, and the like.

Personalization: Analytics-based insights may be used to segment users and provide highly relevant and personalized content, tools, and resources.

CXM may also provide commerce features and optimized integrations with enterprise interfaces.

The key attributes of customer experience are depicted in Figure 4.10:

CMS-based CXM

Let us now look at developing a CXM platform using CMS features. As a matter of fact, most of the modern CXM platforms use CMS as a backbone to

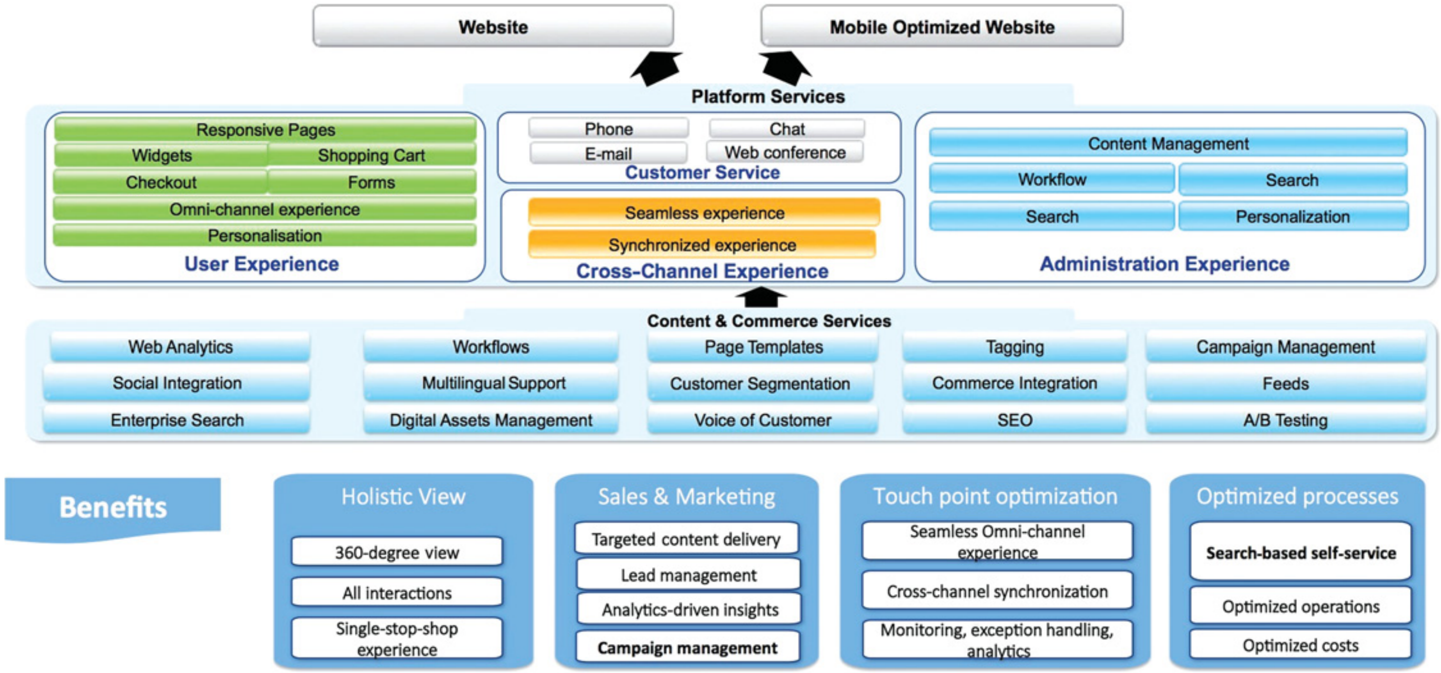


Figure 4.9 CXM Platform Capabilities

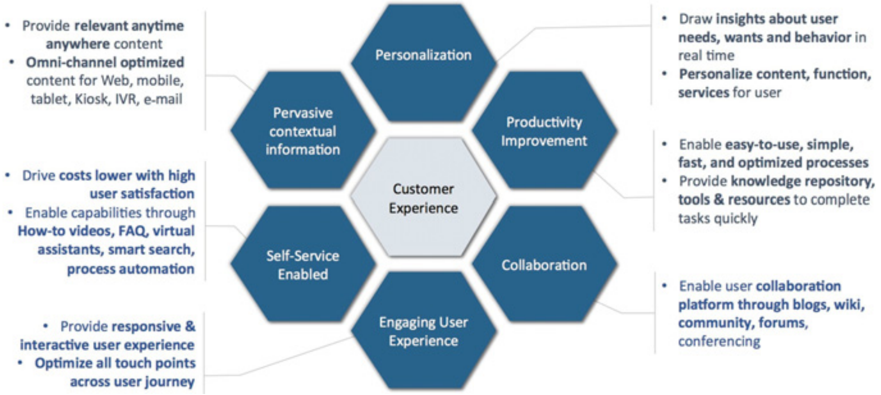

• Provide relevant anytime anywhere content
• Omni-channel optimized content for Web, mobile, tablet, Kiosk, IVR, e-mail
• Drive costs lower with high user satisfaction
• Enable capabilities through How-to videos, FAQ, virtual assistants, smart search, process automation
• Provide responsive & interactive user experience
• Optimize all touch points across user journey
Personalization
Pervasive contextual information
Productivity Improvement
Customer Experience
Self-Service Enabled
Collaboration
Engaging User Experience
• Draw insights about user needs, wants and behavior in real time
• Personalize content, function, services for user
• Enable easy-to-use, simple, fast, and optimized processes
• Provide knowledge repository, tools & resources to complete tasks quickly
• Enable user collaboration platform through blogs, wiki, community, forums, conferencing

Figure 4.10 CXM Attributes

implement the required capabilities. A sample CMS-based CXM platform is depicted in Figure 4.11:

Note: The technologies depicted in the reference architecture are for pedagogical purposes only. It is possible to implement the CXM platform using alternate technologies and platforms.

Let us look at each of the layers in detail.

Presentation layer: The main features in this layer are RWD-driven responsive layouts, media queries, and fluid design using progressive enhancement techniques. As depicted in the reference architecture, this layer uses a forward-looking lightweight model that can cater to all devices. We can create interactive widgets and presentation components. We can implement this layer through technologies such as HTML 5, CSS 3, node.js, backbone.js, and the like. Components in this layer mainly get the backend data in JSON format through REST services.

Note: Some of the CMS platforms provide lightweight presentation components with built-in RWD support. In such cases we can use the native CMS presentation engine.

Services layer: This layer consists of a services handler and an API management platform that uses backend CMS system to retrieve required content. Main tasks handled by this layer are services orchestration, services caching, and abstraction of backend systems.

CMS layer: CMS forms the main engine of this CXM platform. We map the CXM platform capabilities to CMS solution components in Table 4.6.

Interfaces: CMS may be integrated with other systems to provide the required CXM capabilities. In this layer we may have e-commerce engine, analytics engine, campaign tool, ERP systems, enterprise database, external social media platforms, and so forth.

Complimentary Technologies for building CXM

Table 4.7 presents some of the popular technologies and products that can be used alongside CMS technologies for building a CXM.

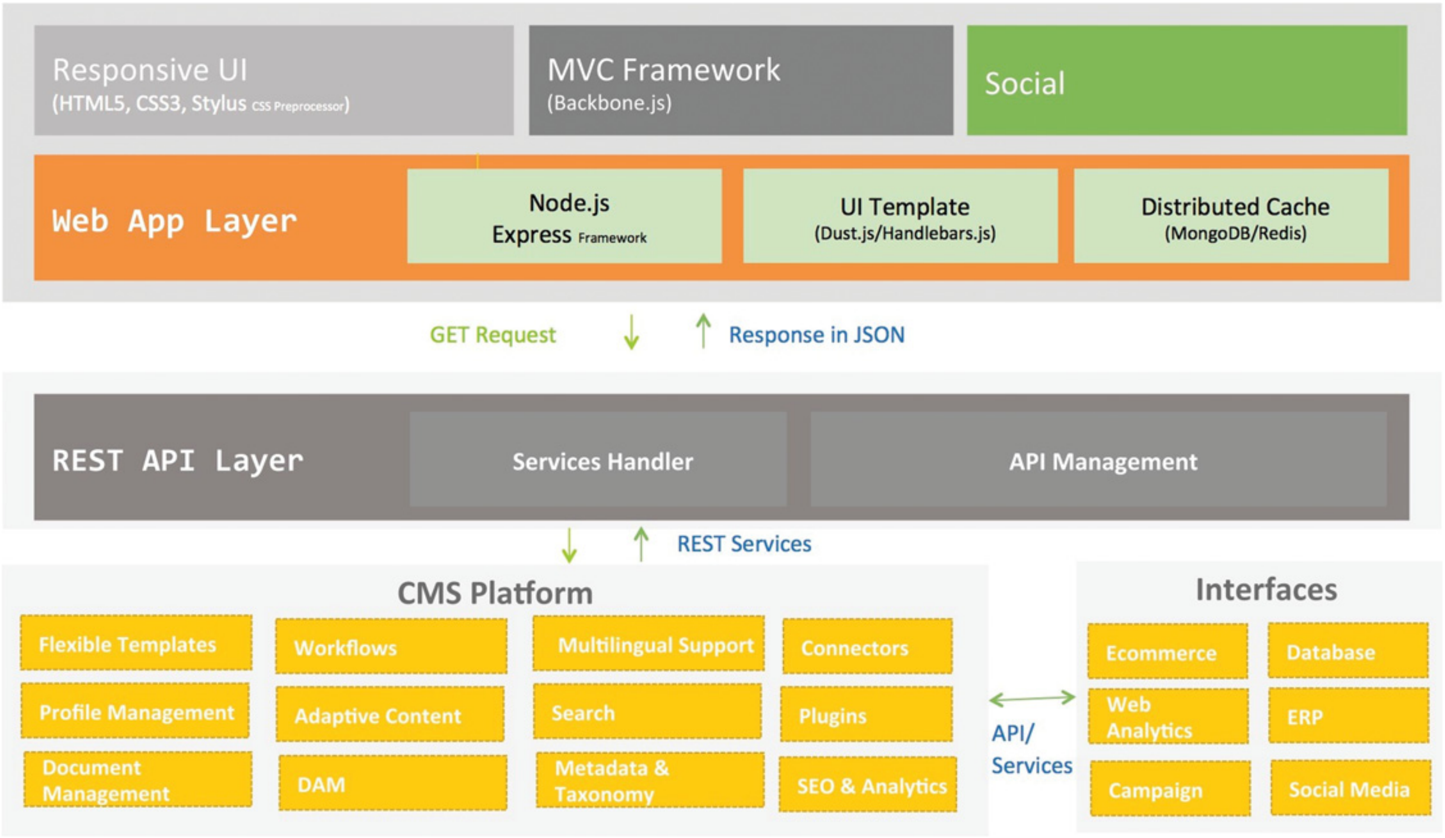


Figure 4.11 CMS-Based CXM Reference Solution Architecture

Table 4.6 CXM capabilities to CMS solution components

CXM Capability	CMS Solution Component
Omni-channel enablement	Adaptive content design can be leveraged for Omni-channel content.
Process optimization	Built-in workflow engine can be used to implement business processes. We can do further process optimizations such as parallel execution, auto forwarding, auto approval, etc.
Analytics-driven insights	Web analytics can be tagged to CMS content to track its usage and user behavior.
Personalization	Built-in user segmentation feature can be leveraged to provide personalized and targeted content.
Information discovery	Built-in content search engine can be used for information discovery. We can also integrate CMS with an enterprise search engine.
Social and collaboration	Most CMS platforms provide collaboration features such as blogs, wiki, community, and forums that can be leveraged. CMS can be integrated with external social platforms using adaptors or through services-based integration.
Marketing and sales engagement	Built-in multivariate testing can be leveraged to provide optimal user experience. CMS can be integrated with campaign tools for developing marketing campaigns.

Note: The list in Table 4.6 is for educational purposes only. The functionality can be realized using alternative technologies and products.

We can also build responsive single-page applications (SPA) using the technologies given in Table 4.6, which can communicate with CMS through REST-based services.

Knowledge Management System Based on CMS

Knowledge management (KM) system is one of the most prominent applications of CMS and search. CMS is ideally suited to create knowledge assets and manage the lifecycle of those assets. Let us have a closer look at various modules involved in CMS-based KM as depicted in Figure 4.12.

Presentation Components

This layer consists of knowledge management user interface components that interact with the end-user. Users can view and update articles based on their access controls. Users can use search to do a faceted browsing of the solution articles, and end-users can also contribute/co-create the solution articles. End-users can provide feedback and rating that will be stored in the CMS.

Table 4.7 CXM technologies

Category	Candidate Products and Technologies	How They Can Be Used along with CMS for Building CXM
MVC Framework	• AngularJS • BackBone • ExJS	These technologies can be used to build a responsive front-end layer/Web app layer and page templates, providing a clean separation of view and model. We can get content from back-end CMS through REST-based content services.
Style Sheets	• CSS3 • LESS	These components can be used to provide reusable classes for high performance.
Responsive Charts	• HighCharts • ExtJS • D3	We can leverage these products to provide highly interactive charting capabilities in the front end.
Templates	• HTML5 • Velocity Template /FreeMarker Template	We can use these components to create flexible templates and build single-page applications (SPA).
Responsive Web Design (RWD)	• Twitter Bootstrap	We can use this to enable RWD in CMS front end to cater to all supported devices and channels.

Content Management Components

This layer consists of various CMS components such as templates, workflows, metadata services, content search, DAM, and others used to manage knowledge operations. CMS components can be used for the following KM features.

KM asset management

The key KM assets are solution articles, FAQ, How-to articles, troubleshooting tips, user guides, product installation instructions, and so forth. Each of the KM assets will have a distinct content model. For instance, a How-to asset consists of video content and summary content. We can define all the necessary content models needed for the knowledge management solution.

We can then create the content-authoring templates for each of these content models and use them to author the content for corresponding KM assets. Content and assets are tagged with built-in metadata to ease discovery.

Document management

A CMS system can act as a content and document repository for the knowledge management system. CMS can use its asset repository to manage documents (uploading,

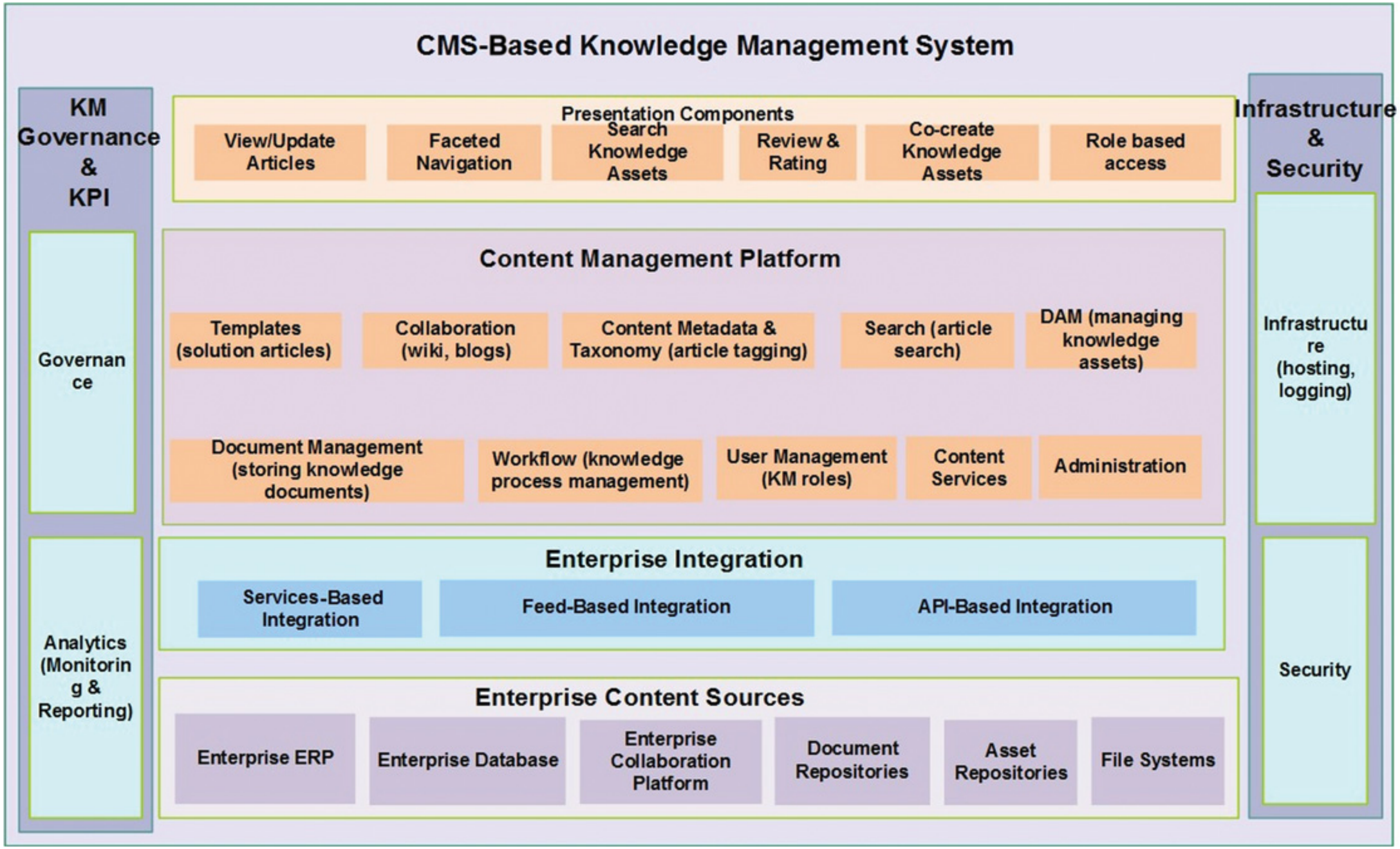

CMS-Based Knowledge Management System
KM Governance & KPI
Governan ce
Analytics (Monitorin g & Reporting)
Presentation Components
View/Update Articles
Faceted Navigation
Search Knowledge Assets
Review & Rating
Co-create Knowledge Assets
Role based access
Content Management Platform
Templates (solution articles)
Collaboration (wiki, blogs)
Content Metadata & Taxonomy (article tagging)
Search (article search)
DAM (managing knowledge assets)
Document Management (storing knowledge documents)
Workflow (knowledge process management)
User Management (KM roles)
Content Services
Administration
Enterprise Integration
Services-Based Integration
Feed-Based Integration
API-Based Integration
Enterprise Content Sources
Enterprise ERP
Enterprise Database
Enterprise Collaboration Platform
Document Repositories
Asset Repositories
File Systems
Infrastructure & Security
Infrastructu re (hosting, logging)
Security

Figure 4.12 CMS-Based Knowledge Management Solution Architecture

storing, updating, and deleting). Documents are tagged with appropriate metadata for effective discovery.

KM users and workflow management

The main KM users are author, reviewer, and publisher. Authoring flow and publishing flow are the key workflows. We can assign the roles of "KMAuthor," "KMReviewer," and "KMPublisher" to existing CMS users and define access controls to the corresponding templates and workflows. We can create a "Knowledge Publishing" workflow to implement the authoring/publishing business rules. If the knowledge management system is accessible across various locales, we also need to create localization workflow to translate the KM assets.

Knowledge discovery

The ability to find the right solution article (KM asset) for a given search term is vital for the success of a knowledge management system. This requires tagging the KM assets with appropriate semantic metadata tags. We can leverage the tagging feature available in CMS for this. The content folder structure and taxonomy in CMS can be leveraged to logically group the KM articles in a hierarchy.

Knowledge search

CMS can be integrated with a search engine to enable the search within a KM system. A search engine indexes content and documents available within CMS and uses them for responding to search queries.

For searching content and assets from the knowledge management system, CMS may support the following content search features:

- Keyword- and metadata-based content search: The query keyword is matched with metadata or tags associated with knowledge assets.
- Guided navigation and faceted content search: In this scenario various content attributes (referred to as "facets") help the end-user filter the content results. Content categories, relationships, and other content attributes can be presented as facets for end-user.
- CMS can provide admin interface for configuring search operation and also support search in its native content object model.
- CMS supports various modes of content queries such as xpath and JCR query language, and also exposes content services that can be used by an external search engine.

Integration of KM with enterprise interfaces

The main modes of integration of KM system with enterprise systems such as ERP, database, document systems, asset systems, and file systems are through services and

APIs. CMS provide built-in support for service consumers, which could be leveraged for this.

Digital Marketing Platform Based on CMS

The overall picture of a digital marketing platform is shown in Figure 4.13. As we can see, CMS forms a key technology enabler for the platform.

CMS can be effectively used for building a robust digital marketing solution. A digital marketing solution needs to host and consume various content formats and present content in various channels. Besides, there is a need for stricter governance and geo-specific business processes and business rules. Most of these requirements can be met by a CMS platform through built-in features such as content repository and workflows.

Digital marketing is one of main use case scenarios for CMS. Figure 4.14 depicts sample CMS architecture for digital marketing function. We can use core CMS features for authoring promotion, campaign content, marketing, and product content. Besides this the CMS authoring and workflows can also be used for building wiki and other knowledge functions. CMS security and connectors can be used for integration with third-party systems.

Table 4.8 depicts the mapping of digital marketing features with CMS solution components that can be used to implement those features.

The features of a digital marketing content are shown in Figure 4.15.

The key feature of digital marketing content is to have a targeted delivery mechanism. Using content metadata, Web analytics tracking insights, and personalization features, relevant content should be delivered to the target audience. Digital marketing content should be personalized based on user segments, target device, and other context parameters. Web analytics should be tagged along with content to track user behavior to gain vital insights. Digital marketing content should be optimized for all devices and user channels. Data and content management best practices should be followed throughout the content lifecycle stages.

Architecture of Apache Jackrabbit

We have seen the general description of Apache Jackrabbit in Chapter 3. In this section we take a look at the key aspects of its architecture.

The general layers of the Jackrabbit architecture are:

- **Content Application Layer** that uses JSR 283 APIs to communicate with repository implementation layer
- **API Layer** that supports JSR 283 features and other features
- **Content Repository Implementation Layer** that contains the key elements of Jackrabbit implementation. The components of the content repository implementation are:

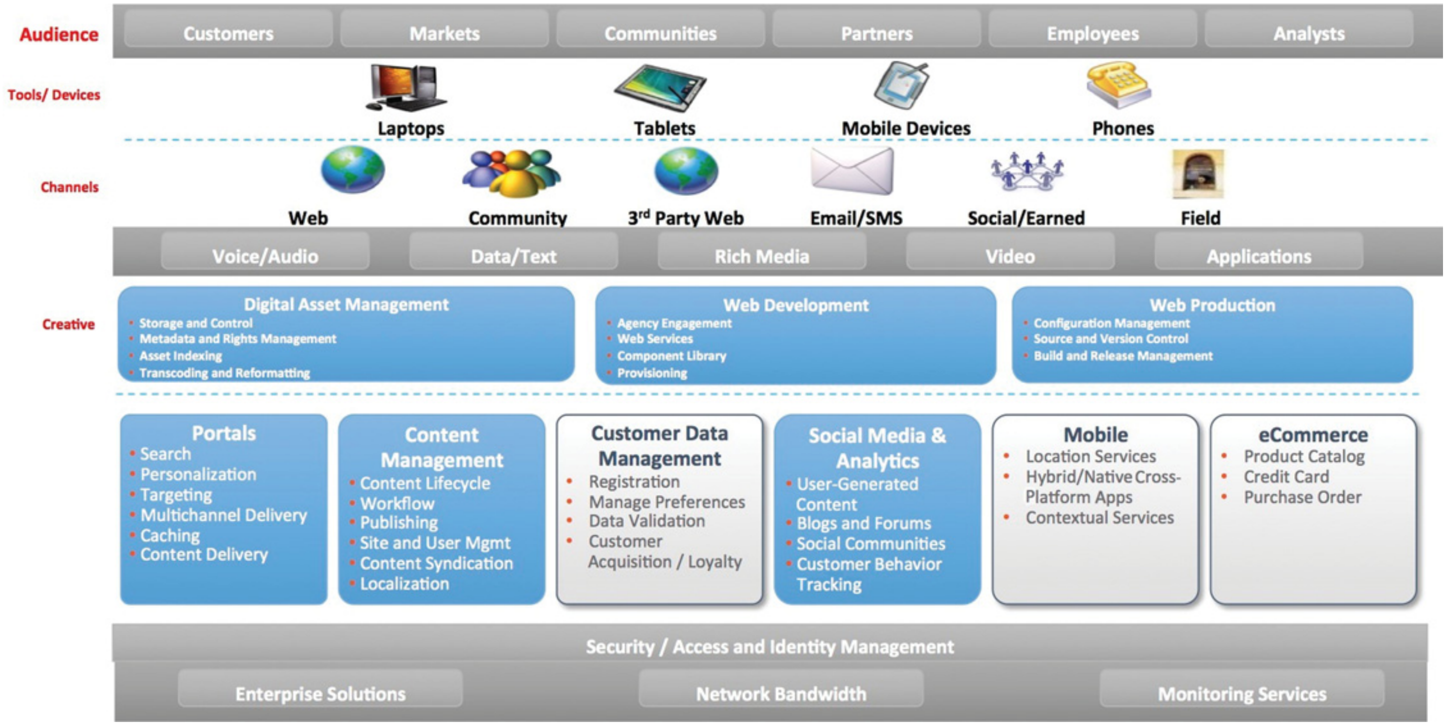


Figure 4.13 Digital Marketing, Overall Picture

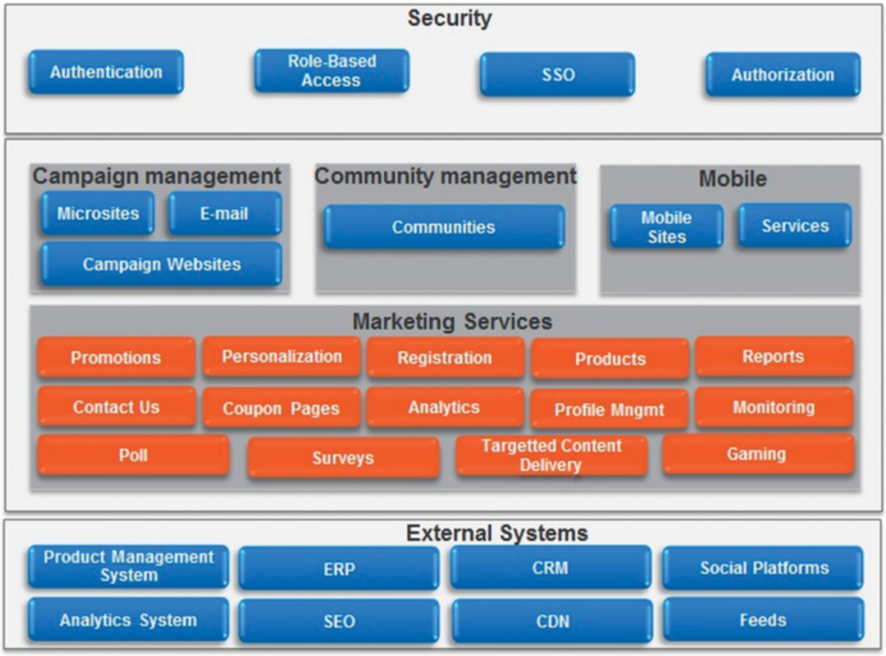


Figure 4.14 Digital Marketing Reference Solution Architecture

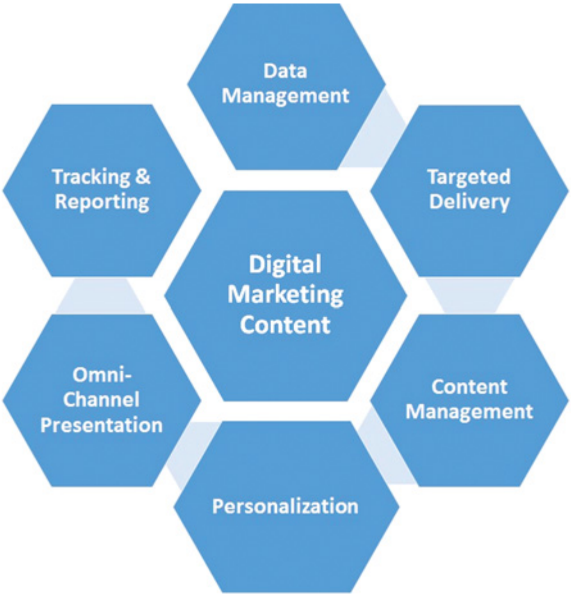


Figure 4.15 Digital Marketing Content Features

Table 4.8 Digital Marketing to CMS solution component mapping

Digital Marketing Feature	CMS Solution Component Details
Multi-channel campaign	• Adaptive content for creating Omni-channel campaign content • Supporting multi-channel capabilities through device rules, multi-language support, device-optimized presentation templates, and layouts • Multi-site management features can support campaigns for various geographies • Rich media requirements can be achieved through digital asset management • Content and asset reuse to support multiple campaign across devices
Personalization and user segmentation	• CMS permission model, role-based access, and metadata can be leveraged to provide personalized delivery • Content analytics can be used to collect details about user behavior and personalize the user experience
Targeted campaigns	• Content analytics can be used to track the usage and create effective targeted campaigns
Micro-site management	• CMS authoring and publishing features can be used to create and launch micro-sites
Promotion management	• Reusable authoring and presentation templates can be used to create promotion content • Use of appropriate SEO tags with content can enhance content and site visibility
Campaign tracking	• Web analytics scripts tagged with content can help in tracking the effectiveness of campaign content
Social campaigns	• Many of the built-in CMS features such as content rating, blog, and wiki can be leveraged for social campaigns • Additionally CMS can also be integrated with external social platforms (through services, plugins, widgets, or APIs) for creating external social campaigns
Polls and Surveys	• The form content for polls and surveys can be built using appropriate authoring templates
Targeted content delivery	• Tag the content with appropriate metadata so that it is possible to deliver the most appropriate content
Collaboration	• Native CMS support for collaborative features such as blog, wiki, communities, and forums can be leveraged to build this feature

- Nodetype
- State
- Session
- NamespaceRegistry
- Repository
- Workspace
- Path
- HierarchyManager
- Version
- Observation
- Xml
- QName
- Query
- ItemManager
- ItemImpl, PropertyImpl, NodeImpl
- ItemId, PropertyId, NodeId

The main layers are depicted in Figure 4.16.

Content Update Process

When user adds or updates content in the content repository, the following steps are executed:

- Transient Item State Manager caches the content items after session reads. The visibility of updated content is only restricted to that user session.
- Transactional Item State Manager retrieves the modified content once it is saved.

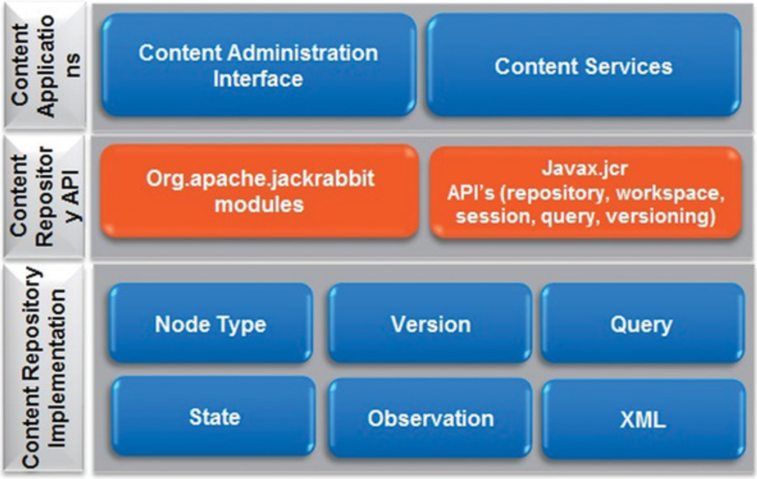

Content Applicatio ns
Content Administration Interface
Content Services
Content Repositor y API
Org.apache.jackrabbit modules
Javax.jcr API's (repository, workspace, session, query, versioning)
Content Repository Implementation
Node Type
Version
Query
State
Observation
XML

Figure 4.16 Apache JackRabbit Architecture

- Shared Item State Manager receives and publishes the changes after the transaction is committed.
- Persistence Manager will persist all the item state that are changed and received by the shared item state manager using the item ID associated with the item.
- Observation module helps the applications to asynchronously get the changes into the workspaces.
- Query Manager/Index module indexes the new items or modified items when the observation event sends an instruction for this action to the Query Manager.

Jackrabbit also has a powerful and flexible search implementation, which can take query parameters in XPath or SQL format.

Note: We have provided sample Java code for adding an image node to Jackrabbit workspace in Appendix E, which compliments the information provided in this section.

4.9 CHAPTER SUMMARY

- CMS selection and implementation methodology consists of discovery and planning phase, definition phase, and development phase
- The discovery and planning phase captures all content requirements; the definition phase defines the CMS solution principles and guidelines and evaluates CMS products; in the development phase we design and develop all CMS solution components.
- CMS value articulation framework maps the program goals to CMS solution components and tracking metrics.
- The main CMS solution design principles are layer-wise separation of concerns, flexible templates principle, functionality fitment and mapping, operations self-service, robust scalability, reliability, content security, extensible and future proof, and standards-based solution.
- The main aim of layer-wise separation of concern is that each layer in the solution should handle distinct responsibility; flexible template design principle aims to create flexible authoring and presentation templates.
- Functionality fitment and mapping design principle aims to map all business requirements into solution components; operations self-service aims to provide independent and self-service model for administrators and business stakeholders; robust scalability aims to make the system scale to handle peak load with optimal response times; reliability aims to make content reliable, and content security provides secured content access.
- Extensible and future proof design makes the application flexible to meet future needs; standards-based solution uses open standards for content development, storing, and integration.
- Multi-site management is needed for globally distributed sites.

- Multi-site design mainly consists of primary site owning core content assets and secondary sites.
- Multi-site deployment strategy consists of live copy approach and partial localization approach.
- Three types of content URL are tree-type/hierarchical URL, flat URL, and dynamic URL.
- Localization design mainly consists of localization requirements elaboration, leveraging CMS capabilities for localization, and localizing content.
- CMS deployment model mainly consists of authoring and publishing instances.
- Release management involves authoring and reviewing on authoring instance and packaging and deploying it into delivery instance.
- CMS maintenance activities include backup, monitoring, and patching/upgrades.
- At the end of the chapter we have seen sample architecture for a knowledge management system and digital marketing systems build using CMS.

Chapter 5

Development Using Templates and Workflows

Templates and workflows are the quintessential components for content design. Templates provide a reusable and consistent framework for managing content. Authoring templates enable authors to create content with defined structure. Presentation templates add presentation logic to content. Content workflows play a vital role in the content management lifecycle. Processes such as content publishing, content translation, and content versioning are modeled by content workflows.

In this chapter we will look at various aspects of content templates and workflows, along with best practices and optimization methodologies. We will begin with basic concepts of templates, template types, and details of authoring templates and presentation templates. We will then look at template-user interfaces and mapping templates to Web pages. We also will look at case studies for designing a template for a Web support site. In the next part of the chapter we will look at workflow design and workflow optimization along with a case study.

Content architects, content authors, and CMS developers will find this chapter useful.

5.1 CMS TEMPLATE DESIGN

Templates are the core CMS solution components used for content authoring and publishing. Templates provide structured framework components for content design and development. Templates play a key role in efficiently managing the content processes. In this section we look at various aspects of template design, template development process, template best practices, and a case study.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

What Are Templates?

A CMS template is essentially a reusable component that provides authoring structure for content. Templates provide uniform and consistent structure for Web content. Templates aid content authors and editors by providing the predefined layout and structure for the content and capturing all the required information. Authors can select the most appropriate template and start adding content to it. While most CMS provide built-in templates, they also allow the administrators to create custom templates to suit the specific business requirements.

Content strategy provides vital inputs for designing a template. The unique content structures, content models, and content types gathered by a content strategist are used to create a flexible template.

Authoring Template, Presentation Template, and Page Layout

It is important to outline the differences between the authoring template, presentation template, and page layout before discussing remaining sections. ***Authoring templates*** are components for authoring content for Web pages. Authored content may or may not contain presentation logic (depending on the rendition workflow). The recommended option is to store core content without presentation logic (in form of XML, JSON, etc.).

Presentation templates add the required style elements to core content. They add the required presentation logic (such as HTML elements, layout elements, styles, formats, scripts, presentation rules, etc.) to core content. Having a separate authoring template and a presentation template helps reuse content across various presentation channels and devices. Both authoring and presentation templates can be used for entire page content or for smaller content fragments (also referred to as content chunks).

Page layout defines the structure of the end Web page. Normally CMS provides predefined page layouts, which can be customized to suit specific needs. An authoring template facilitates creation of content to suit the end Web page. Each unique page layout may require a different content structure, and an authoring template may help create content with that structure. Presentation templates transform core content (authored using authoring templates) into a structure that can be rendered onto end Web pages. In a content chunk scenario, an end Web page consumes content generated from various authoring templates (which generates content chunks).

Design considerations for an authoring template

The main design considerations while designing a template are as follows:

- A template should be flexible to support content of all granularity. High-granular content creates multiple content fragments/chunks for a page, whereas low-granular content creates fewer content fragments for a Web page. Templates should be able to support both these types.

- Flexibility and reusability are the key design considerations while designing a template. The structure of the template should support reusability and creation of varied content structures (content models) to make it more versatile. This can be done through optional template elements, repeatable sections, and other flexible features. Separating core content from its end presentation also greatly enhances its reusability. New custom templates need to be created only when the built-in templates cannot be used.
- Templates should allow tagging of metadata and SEO elements.
- Templates should also allow the author to add Web analytics scripts and tags.
- Templates should also allow users to specify the language, saving to a particular destination.
- Templates should help authors associate content with a defined workflow.

Developing an authoring template An authoring template is the core component of a CMS, so lot of due diligence needs to be done before developing one.

Prerequisites for developing a content-authoring template

Before developing a template, the following activities must have been completed:

- Understand all unique page layouts required for the solution. For instance, home page content may need a three-column layout, whereas a product description page may require a two-column layout. So we need to compile the inventory of all unique page layouts needed for the solution. This will be used for creating templates for creating page-level content.
- Identify all unique content structures/content model/information model required for the solution. A content model/information model specifies the structure of content and provides the elements within a given content chunk. For instance, a product short description information model may contain a product thumbnail image, label, and a brief product description (preferably not exceeding fifty words); a promotion event content model includes a content section and a video section; FAQ content type includes just a single content section; a banner content model includes an image section and a content section. Templates help us create content for a particular content type by providing the required elements.
- Identify all the attributes needed for a content model.
- Identify the data types for each of the attributes in the content models (such as String, text area, blog, HTML, image, document, etc.)

- Identify the appropriate content granularity for the Web site. In this step we identify the approximate percentage of reusable content chunks/fragments required.
- Once we have a list of all available page layouts, content models, attributes and content granularity, we can design the template considering the following factors:
  - In some CMS platforms, page templates and authoring templates form a single entity (though this impacts content reusability). In such cases, we need to identify the most minimal set of unique layouts, which can be used to render all the required page structures. For instance using a two-column page layout can render the product-landing page and product detail pages, so we just need one distinct page structure/layout to render these two set of pages. This helps us create the distinct library of content structure, which can then be converted into content templates. This makes the content templates reusable across a wide set of pages.
  - Templates should be designed to support all content models. This requires having the required fields/placeholders (such as fields to specify title, image selectors to attach the image, RTE to create rich text, etc.) to create the required content models.
  - Templates should be able to create content of all granularity. This means that there should be templates to create the large content chunk (which can become a complete Web page) and smaller content chunks (which are reused across multiple pages).
- CMS should be integrated with any digital asset management system for attaching assets while authoring content in template.
- CMS should be integrated with metadata system or the metadata hierarchy should be defined within CMS.
- Wherever required, CMS should be integrated with a translation management system (TMS), and content workflows should be defined.

Design of Authoring Templates

Content-authoring templates are designed and developed to support all the information models required for the enterprise. Most CMS systems provide tools for creating the templates. Let us look at the steps in creating a custom-authoring template.

Step 1: Create template elements. A typical content-authoring template contains the following elements:

- A title to store the content title
- Required content identifiers such as name and brief description required by the content model
- An asset section to which we can attach any static asset from the DAM system or from internal document repository. Normally there would be

a file upload feature or asset drag-and-drop feature to attach the asset to content.

- A link where we can add document links or content hyperlinks
- Keywords where we can specify the content keywords. This will be used for SEO purposes.
- Content sections, which allow authors to add content. A template can have multiple content sections, subsections, and nested sections based on the complexity of the content structure. We can enhance the flexibility of the template through optimal use of these content sections, such as:

  a. Allowing multiple optional repeatable content sections

  b. Allowing nested content sections

  c. Allowing reordering of content sections

- A highly flexible set of content sections (sometimes called containers) make the template more reusable because it makes it possible to author a wide variety of content structure of various complexities. This element usually provides rich text editors (RTE) and WYSIWYG editors for specifying content.
- Tags where we can add metadata tags for content sections
- A disclaimer where we can add any optional disclaimers applicable for the content
- A location where created content will be stored
- In addition to these core elements, a template may also support additional elements related to language, content category, analytics tags, SEO tags, etc.

Step 2: Provide an option to associate the workflow for the template.

Step 3: Provide an option to choose the presentation template. This will be used for rendering a template for a particular presentation platform.

Step 4: Provide an option to specify the site area or section where the authored content will be rendered.

Step 5: Assign permissions to the template. This restricts the users that can access the template.

Some CMS may also allow templates to provide a content preview feature, allowing authors and editors to review the changes.

Content Templates Based on Open Standards

While it is possible to create templates using CMS provided tools, it is also possible to create the templates using open standards/technologies such as XML, Velocity, and Free Marker templates.

An XML schema document can be used to specify the constraints and enforce rules of a template. We can build a user interface that can enforce these constraints.

Some CMS platforms provide built-in support for these open standards. For instance, in Liferay CMS, we can build WCM templates based on Velocity and FreeMarker templates along with their macros. This helps leverage the powerful features offered by these frameworks for data iteration, dynamic variable substitution, and other functions. Velocity templates are widely used as e-mail templates.

Content Presentation Templates

In addition to content-authoring templates, CMS would also provide content presentation templates. Presentation templates inject the presentation logic into core content. While the authoring templates help authors create core content, presentation template adds the presentation styles, HTML, and presentation formats to create the overall look and feel of authored content. This two-template model neatly separates content from its presentation. This also makes content delivery flexible, as we can attach various presentation templates based on the target delivery system. For instance, we can create two presentation templates, one each for desktop browser and for mobile browser. Without changing core content, we can present it optimally on different delivery channels.

Presentation templates primarily inject the following presentation elements into the content:

- Presentation formatting elements such as look and feel, content/page layout elements, HTML tags, etc.
- Style elements such as font specifications and color scheme that align with the approved visual style guide specifications
- Height and width dimensions for content
- Presentation validation rules such as show/hide rules based on target device specifications.

A simple presentation template can be implemented using XSL that adds the presentation logic to the XML content.

Content page template vs. content chunk template

Content authoring and presentation templates can be used to author content for an entire Web page or for smaller site sections (through content chunks). Table 5.1 presents scenarios in which a page template or a chunk template can be used.

Table 5.1 Content Page Template and Content Chunk Template

	Content Page Template	Content Chunk Template
Usage scenario	For authoring entire page content	For authoring individual page sections through content chunks
When to use	Can be used to author static pages for which content rarely changes, such as FAQ page, Contact Us page	Can be used to author frequently changing, dynamic content, such as news headlines section, featured products section, popular products section, etc. When a content section/fragment is reused across pages, it becomes an ideal candidate for content chunking
Performance optimization measures	Can cache the entire page content at a Web server or a CDN layer	Can cache the page fragment at a Web server Can load the content chunk details asynchronously
Reusability	Relatively low	Relatively high, as it can be used across pages and site sections

5.2 AUTHORING CONTENT USING AN AUTHORING TEMPLATE

While creating a new page content for the Web site, we need to choose the closest matching template that can be used to author content for that page. If there are no matching templates, then we can use flexible templates. General steps for creating content for a new page are as follows:

- **Page content structure identification**: Identify the structure of the new page content. Page content structure translates to one of the content models. Page content structure may contain various elements such as title, brief description, long description, thumbnail images, and others.
- **Choosing a template**: Once the page structure is identified, find the closest matching template from the template library. A template can match the page content structure if it fully supports all elements required for the content structure/content model. Listed below are the main scenarios:
    - In the best-case scenario we get a perfect match between the template and the new page structure. In this case we can map the template for the new page and use it for authoring and rendering purposes.

  - There could be minor variations between the new page structure and an existing template. For such scenarios we can use the extensible features of the template (such as repeatable sections, flexible paragraph sections, etc.) to author the content. We can also try to leverage the flexible template.
  - Other possibility is that the new page layout is a brand new structure that is not present in the template library. If other pages will use the new page structure, then it would be meaningful to create a new template for it.
- **Author content using best-matched template:** Use the chosen template to author content. The selected content template should be able to support all fields for the target content model/content structure and should be able to create content of required granularity. If we adopt "flexibility" as one of the key tenets of template design, most of the templates should be able to support the fields required for creating various content models.
- **Security permissions**: Once the template is chosen for authoring, we can assign the page permissions for authoring, reviewing, and publishing.
- **Workflow assignment**: Assign the applicable workflows for the new page content. This includes workflows for review and publishing scenarios

Once content is authored, it can be stored in a standard format without presentation elements. In such a case we can use presentation templates or transformation engines to create the end-presentation for the target device. The presentation template adds the format, styles, layout, and other presentation elements to core content.

If there is only one presentation platform (such as the one used by desktop browsers), then the authoring template can provide rich text editing option to add HTML elements, and publishing workflow can publish in HTML format. However, it is always recommended to separate core content from presentation, as it offers more flexibility.

Note: It is important to note the difference between page content structure and page layout structure. Page content structure represents the elements required for a content piece and is represented by page model; it is mainly a content concern. For instance, the content structure/content model for "Product description" content chunk would contain elements such as "product image," "product title," and "product brief description." When we map this to an authoring template, the template should support these three fields.

Page layout, on the other hand, is a presentation concern. In the above example, "Product description" content chunk may be represented in a two-column layout on the Web page (and single-column layout for a mobile platform). In such a case we need to use a presentation template or XSL transformation engine to transform the core content (output from authoring template) into the desired content layout.

Some CMS platforms offer WYSIWYG templates that mix both content and presentation concerns. We can design the page layout using templates and author content for particular sections directly in those templates. In such cases, templates represent the end-presentation. Though this approach seems convenient initially, it is less elegant than the two-template approach is. We may end up creating numerous templates *with minimal content reusability. For simple scenarios (with fewer pages/less content and with only a single presentation platform), we can use the single-template approach.*

Template-User Interface

Let us look at a sample content-authoring template and sample content authored from the template. Figure 5.1 depicts a content-authoring template. We will call it a "flexible section template."

The template mainly has the following sections:

- Title section to author the content title and summary
- Repeatable sections to author individual content sections. We can also create subsections within a section. Each section has title and text. As the template provides RTE for the content text, it can be used only for HTML publishing. For each section we can add metadata tags, as well as nested sections and subsections.
- Optional image section to attach image and specify alternate text and caption
- Keywords section to specify the content keywords

The template is flexible in multiple aspects:

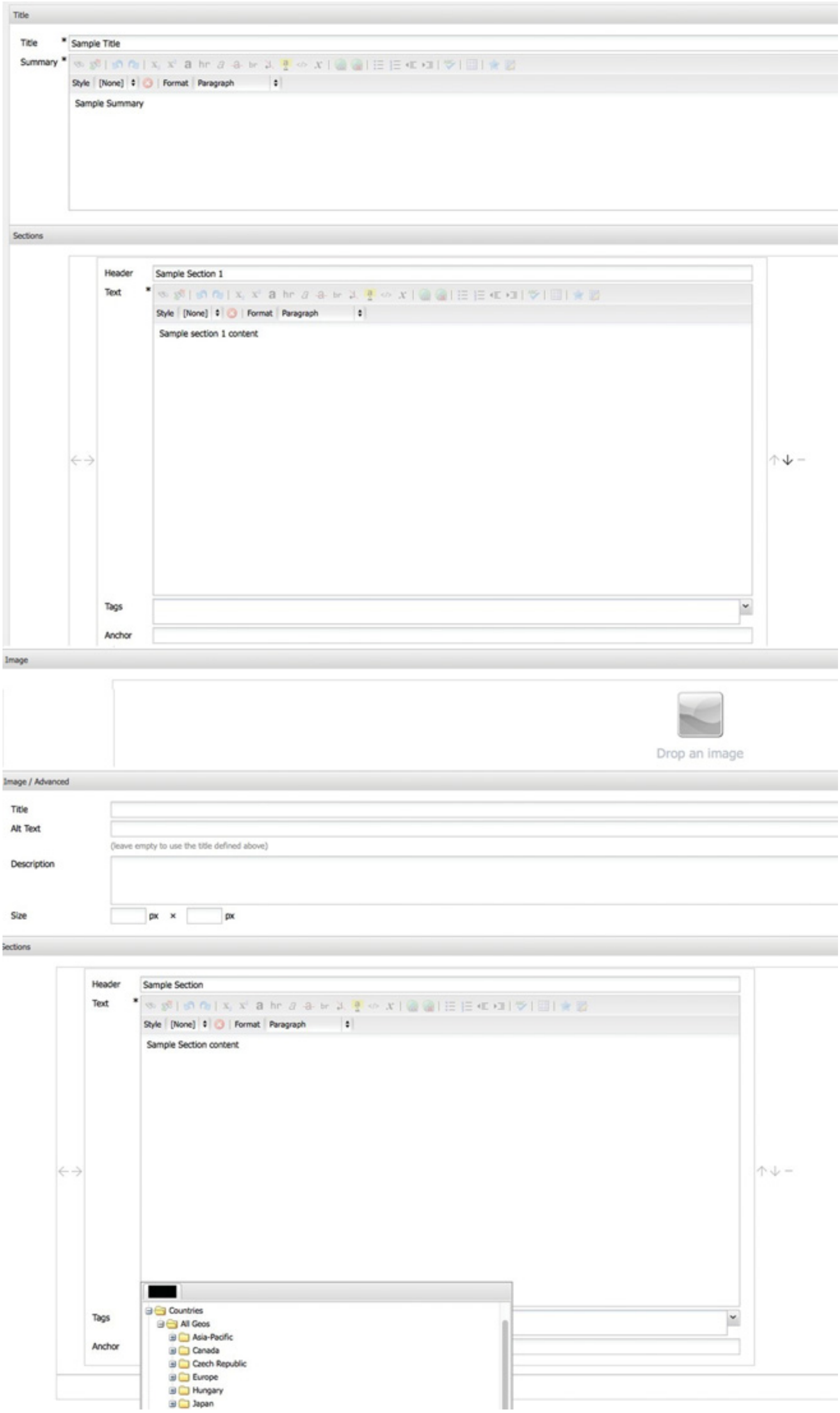
- We can use it for authoring content for an entire page (by adding the required number of sections through the repeatable sections feature) or for a content chunk (by using a single section).
- The template can also support a wide variety of content structures; we can create a simple content chunk with just a heading or we can create a complex content structure by using optional and repeatable sections and an optional image element.
- Complex content structure can be created using nested section and the repeatable subsection feature.

Content authored from the flexible section template can be published as JSON or XML. A sample JSON for a "Product Overview Chunk" is given in Figure 5.2

As we can see from the published JSON, we can specify various nested sections (such as section-1-1) and images (such as image-1), which makes the flexible section template a really powerful tool.

Using Templates for Pages

Let us look at the Web pages that can be created using the flexible section template. Sections of the Web page are identified with template fields. Using presentation templates and transformation engines, we can create a variety of user interfaces for authored content. What follows are two examples of presentation content created



Title * Sample Title

Summary * Style [None] Format Paragraph

Sample Summary

Header Sample Section 1

Text * Style [None] Format Paragraph

Sample section 1 content

Title

Alt Text

(leave empty to use the title defined above)

Description

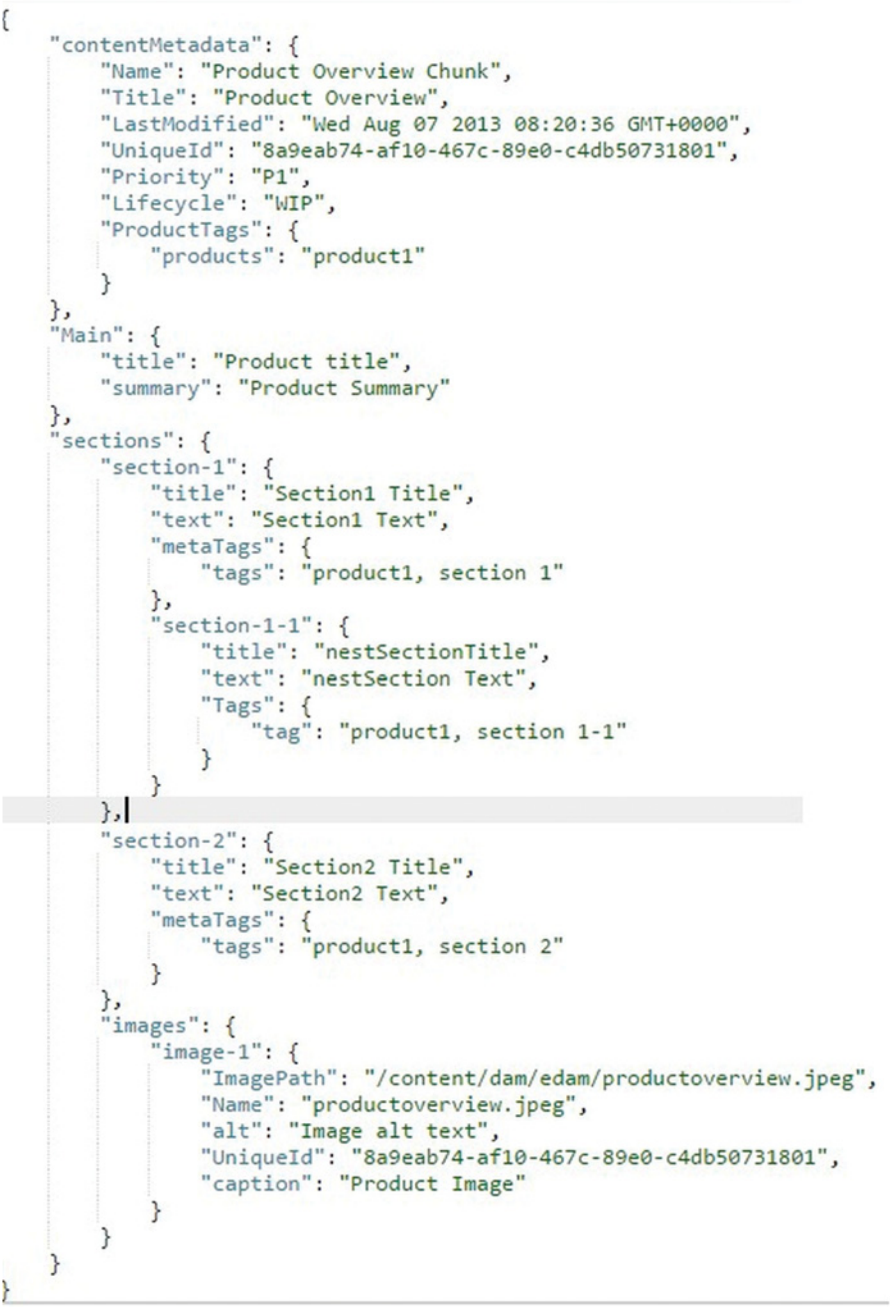
Size px × px

Header Sample Section

Text * Style [None] Format Paragraph

Sample Section content

Figure 5.1 Flexible Section Template



```
{
    "contentMetadata": {
        "Name": "Product Overview Chunk",
        "Title": "Product Overview",
        "LastModified": "Wed Aug 07 2013 08:20:36 GMT+0000",
        "UniqueId": "8a9eab74-af10-467c-89e0-c4db50731801",
        "Priority": "P1",
        "Lifecycle": "WIP",
        "ProductTags": {
            "products": "product1"
        }
    },
    "Main": {
        "title": "Product title",
        "summary": "Product Summary"
    },
    "sections": {
        "section-1": {
            "title": "Section1 Title",
            "text": "Section1 Text",
            "metaTags": {
                "tags": "product1, section 1"
            },
            "section-1-1": {
                "title": "nestSectionTitle",
                "text": "nestSection Text",
                "Tags": {
                    "tag": "product1, section 1-1"
                }
            }
        },
        "section-2": {
            "title": "Section2 Title",
            "text": "Section2 Text",
            "metaTags": {
                "tags": "product1, section 2"
            }
        },
        "images": {
            "image-1": {
                "ImagePath": "/content/dam/edam/productoverview.jpeg",
                "Name": "productoverview.jpeg",
                "alt": "Image alt text",
                "UniqueId": "8a9eab74-af10-467c-89e0-c4db50731801",
                "caption": "Product Image"
            }
        }
    }
}
```

Figure 5.2 Product Overview Chunk

using the flexible section template. These examples show how the template can be used for full-page authoring and for content chunk authoring.

Figure 5.3 shows an example of “Grid layout” content wherein we can lay out various content elements in a horizontal or vertical grid. This layout can be used for product listing, topic listing, and the like.

In the example given in Figure 5.3, a single authoring template can author almost the entire page’s content (with the exception of the header, menus, and left navigation).

The second example, in Figure 5.4, is a home page. This is a more complex scenario, in which we have content in various formats and types. In this case we can use the flexible section template to author individual page sections (via content chunks)

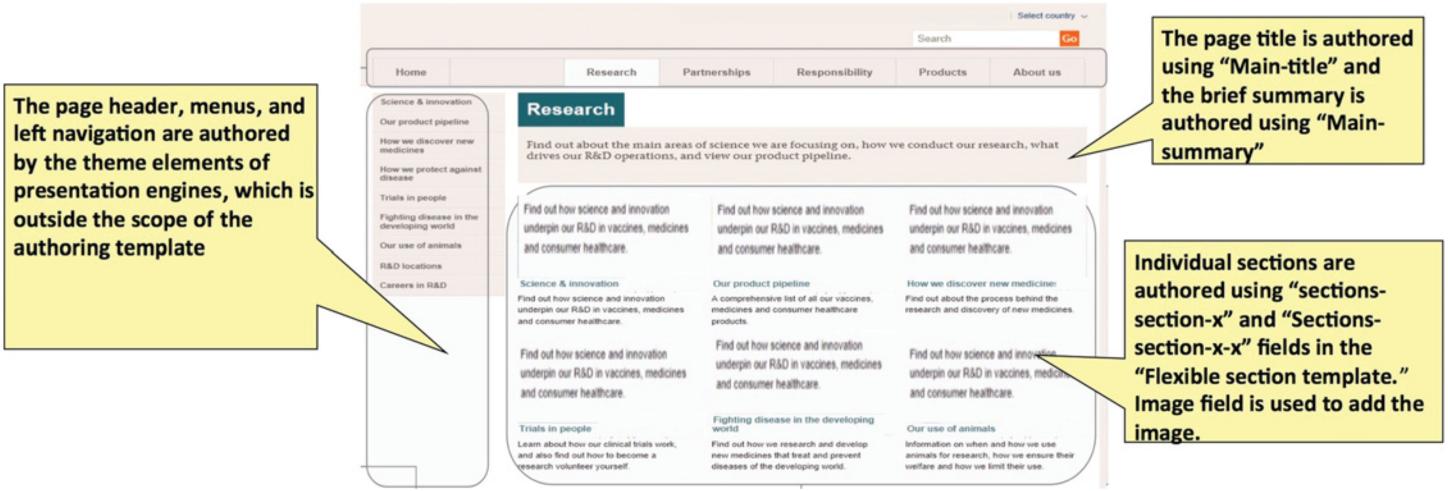
In this example, the flexible section template can be used to author four content chunks (company news, events, leadership corner, and did you know). The rest of the sections are rendered by widgets.

Content chunks form an important aspect of content reusability. We discuss them in greater detail in the next section.

5.3 CHUNKING AND TEMPLATES FOR CHUNKS

One of the effective techniques to enhance content reusability is by using “chunking,” whereby a large content blob is aggregated using multiple content chunks (and conversely large, monolithic content is broken down into modular, reusable content chunks). *Each content chunk is a self-contained, independent logical entity that can be used for creating a topic in a structured way.* For instance, on a product details page, we can have content chunks for product brief description, product features, and product long description. This may help us reuse the individual content chunks on similar pages. The main advantages of the chunking strategy are:

- **Reusability:** Individual chunks can be reused across various content pages and contexts, thereby reducing content duplication and enhancing author productivity.
- **Collaborative authoring:** Allows authors to work simultaneously on various chunks related to same subject matter (one author can author the product description along with second author who can author the product features chunk).
- **Single-source publishing:** Individual content chunks can be published in various formats on various devices. This allows to maintain a single code base and reuse it across various presentation channels.
- **Easier maintenance and faster time to market:** Due to reduced content, the chunks are easier to manage. With chunks we can quickly update page sections, reducing the time to market.
- **Smaller cost and improved productivity:** The cost of authoring, publishing, and managing may be reduced and chunks also may improve the overall productivity.



The page header, menus, and left navigation are authored by the theme elements of presentation engines, which is outside the scope of the authoring template

Select country

Search Go

Home | Research | Partnerships | Responsibility | Products | About us

Science & innovation

Our product pipeline

How we discover new medicines

How we protect against disease

Trials in people

Fighting disease in the developing world

Our use of animals

R&D locations

Careers in R&D

Research

Find out about the main areas of science we are focusing on, how we conduct our research, what drives our R&D operations, and view our product pipeline.

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

Science & innovation

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

Our product pipeline

A comprehensive list of all our vaccines, medicines and consumer healthcare products.

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

How we discover new medicine:

Find out about the process behind the research and discovery of new medicines.

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

Trials in people

Learn about how our clinical trials work, and also find out how to become a research volunteer yourself.

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

Fighting disease in the developing world

Find out how we research and develop new medicines that treat and prevent diseases of the developing world.

Find out how science and innovation underpin our R&D in vaccines, medicines and consumer healthcare.

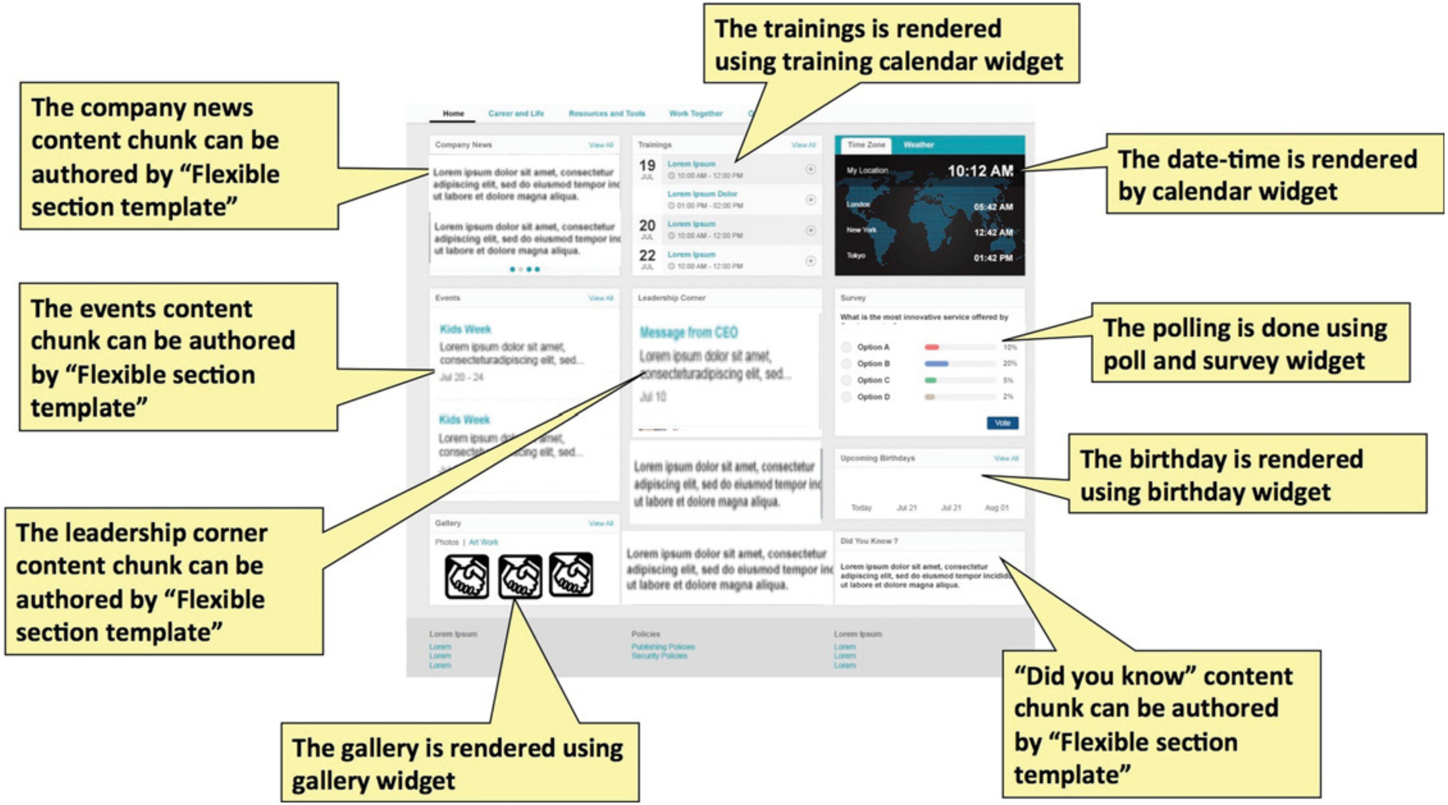
Our use of animals

Information on when and how we use animals for research, how we ensure their welfare and how we limit their use.

The page title is authored using “Main-title” and the brief summary is authored using “Main-summary”

Individual sections are authored using “sections-section-x” and “Sections-section-x-x” fields in the “Flexible section template.” Image field is used to add the image.

Figure 5.3 Full-Page Content Using Flexible Section Template



The company news content chunk can be authored by "Flexible section template"

The trainings is rendered using training calendar widget

The date-time is rendered by calendar widget

The events content chunk can be authored by "Flexible section template"

The polling is done using poll and survey widget

The birthday is rendered using birthday widget

The leadership corner content chunk can be authored by "Flexible section template"

"Did you know" content chunk can be authored by "Flexible section template"

The gallery is rendered using gallery widget

Figure 5.4 Content Chunk Authoring Using Flexible Section Template

Design Considerations for a Content Chunk

We need to consider various factors for identifying chunking candidates. The key design considerations are:

- **Right Granularity:** Too low granularity (such as at field level) may create numerous content chunks and poses the risk of an increased management overhead; similarly, if the chunking is done at a higher level of granularity (such as at a page level), it may impact chunk reusability. Hence, an optimal content chunk should strike the right balance for optimal reusability. During content analysis, a content strategist needs to understand the right mix of content granularity and reuse potential.
- **Chunk candidate identification:** To create a right-sized chunk, we first need to identify a logical piece of modular and self-contained content that can be potentially reused. Another technique to identify the chunking candidate is to look out for duplicate information across various content scenarios. We can then extract the duplicate information as an independent content chunk. We can add other information (such as chunk title, brief description, etc.) to make it logically independent.
- **Metadata and tagging:** Each content chunk should be considered equivalent to regular content. It should be tagged with relevant metadata, SEO, analytics, and other tags (such as target audience, language, application-specific keywords, geography, target device) that help in unique identification and selection of the chunk. Chunk-level metadata is important in selecting the right and relevant chunk and for filtering chunks based on search criteria.
- **Content Model:** Content models should be designed to create content chunks.
- **Chunk templates:** As the chunks can be independently authored, we need to design the authoring templates for the chunks as well.
- **Chunk XSD and DTD:** The XML schema (XSD) or document type definition (DTD) can be used for content chunk XML to enforce the field-level restrictions and applicable business rules.
- **Chunk security:** Content chunks may contain confidential information, and hence permission model should be designed to enforce security and prevent accidental information disclosure. We can leverage the permission model of the native CMS or ensure security via security metadata.

Content Chunk–Based Page Content Aggregation

A chunking strategy provides us with an opportunity to dynamically assemble the page content (or larger content sections) from smaller, reusable modular content chunks. Dynamic content aggregation usually happens at run time based on the context (user preferences, metadata, and site section). Dynamic content aggregation has a multifold impact on the content management process: it not only reduces content

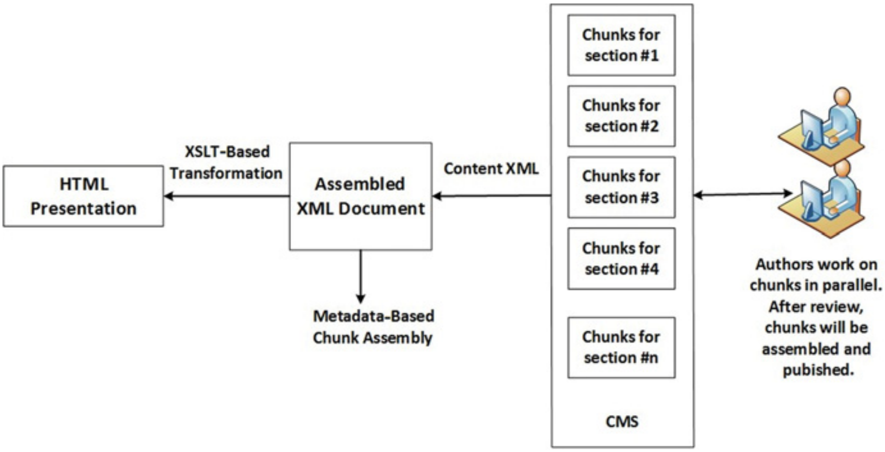


Figure 5.5 Chunk-Based Dynamic Page Aggregation

authoring/reviewing/publishing/testing effort but also may help in dynamic auto-creation of pages (the entire page in the best-case scenario). Chunk-level metadata acts as glue in assembling the relevant content chunks. A sample chunk aggregation is shown in Figure 5.5

In the scenario depicted in Figure 5.5, we have multiple authors creating content chunks for different page sections simultaneously. Authors add metadata tags for content chunks with a page name and a section name. Content chunks are stored in XML format in CMS and then are dynamically assembled based on tagged metadata (such as page name and section name) into a single XML document. The aggregated XML document is transformed into an HTML page using the XSLT engine.

Chunks can be aggregated at two levels:

- **Content authoring template level (Design time content assembly):** We can build the “auto-suggest” smartness into authoring templates, which will suggest the matching content chunks based on the specified metadata/tags and content category. Authors can select the applicable content chunks to speed up the authoring. For instance, if authors are creating a solution article for troubleshooting solution, before authoring the required “prerequisites” for the solution article, the author can provide the applicable keywords (such as “installation prerequisites,” “product family,” etc.), and the template can internally search for all matching prerequisite content chunks and suggest them to the author. The author can then select the most appropriate chunk(s) and proceed to specific troubleshooting steps. This drastically eases the content-authoring effort.
- **Content presentation template level or page level (Run time dynamic content assembly):** At the page level, a presentation engine (such as portal) or a transformation engine (such as XSLT-based engine) can dynamically assemble the relevant content chunks based on metadata. In this case the

dynamic assembling engine would use contextual information to filter out the relevant content chunks. For instance, the presentation engine can select all the content chunks tagged for home page for a specific target audience. The presentation engine can use the user session information to get the filter criteria and personalize the experience.

A sample chunk-authoring template and the chunk rendition are depicted in Figures 5-6 and 5-7. The chunk template can be used to author the disclaimer chunk that will be reused on all pages.

Case Study: Chunk Identification and Chunk Template Design for Product Pages

This section provides a detailed case study that elaborates on how to identify a chunk from a given page and design the chunk templates for it. We examine template design concepts in addition to the chunk design elements in this case study. We also take a look at product pages for a retail Web site. Product pages provide details about various products and their features.

We begin with a look at sample chunks for various product pages. Figure 5.8 is a product category page.

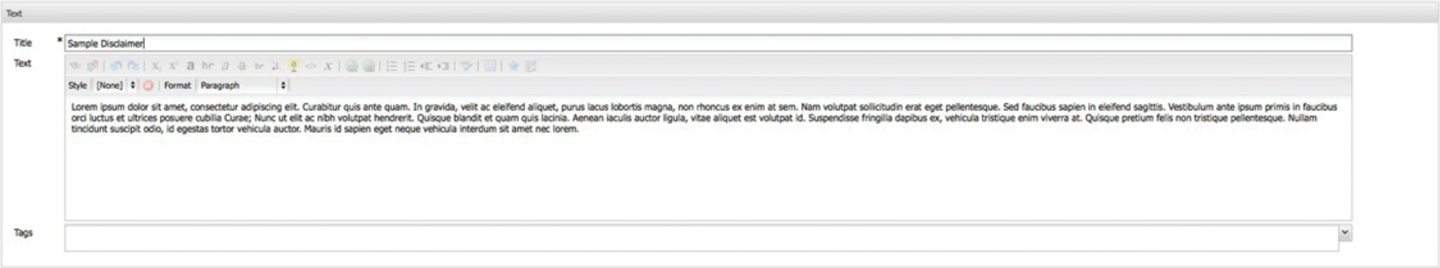
Chunk template identification We will use the main design considerations we discussed, such as independence, reusability, modularity, logical entity, and self-contained attributes, to identify and extract the chunks. For each of the identified chunks we will have a corresponding content model and content template.

As an initial step, we identify the reusable content chunks. The top marquee section is common across all pages, and hence we can create a chunk template for that; we will call it the "marquee chunk template." Similarly the right-hand boxes (Right-Hand Section 1, Right-Hand Section 2, and Right-Hand Section 3) represent an independent logical entity, reusable across various pages. This qualifies the right-hand section elements for a chunk; we will call it the "right-hand section chunk template."

The structure of main product category information is consistent across all product category pages. Thus we can create a chunk template to hold this information and call it the "product category chunk template."

In addition to these three chunk templates, we will also create a flexible chunk template that can be used for all scenarios. Basically we can use "*flex section chunk template*" to author content of any structure. In above scenario we can use flex section chunk template to author language selector chunk.

Chunk templates development The chunk templates on the list that follows have been identified earlier. We will be using XML-based templates for this case study. XML-based templates can also be easily converted into authoring templates supported by CMS.



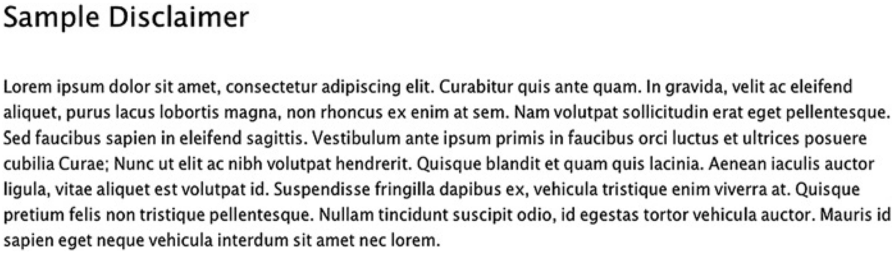
Title * Sample Disclaimer

Text

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur quis ante quam. In gravida, velit ac eleifend aliquet, purus lacus lobortis magna, non rhoncus ex enim at sem. Nam volutpat sollicitudin erat eget pellentesque. Sed faucibus sapien in eleifend sagittis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nunc ut elit ac nibh volutpat hendrerit. Quisque blandit et quam quis lacinia. Aenean iaculis auctor ligula, vitae aliquet est volutpat id. Suspendisse fringilla dapibus ex, vehicula tristique enim viverra at. Quisque pretium felis non tristique pellentesque. Nullam tincidunt suscipit odio, id egestas tortor vehicula auctor. Mauris id sapien eget neque vehicula interdum sit amet nec lorem.

Tags

Figure 5.6 Disclaimer Chunk Authoring Template



Sample Disclaimer

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur quis ante quam. In gravida, velit ac eleifend aliquet, purus lacus lobortis magna, non rhoncus ex enim at sem. Nam volutpat sollicitudin erat eget pellentesque. Sed faucibus sapien in eleifend sagittis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Nunc ut elit ac nibh volutpat hendrerit. Quisque blandit et quam quis lacinia. Aenean iaculis auctor ligula, vitae aliquet est volutpat id. Suspendisse fringilla dapibus ex, vehicula tristique enim viverra at. Quisque pretium felis non tristique pellentesque. Nullam tincidunt suscipit odio, id egestas tortor vehicula auctor. Mauris id sapien eget neque vehicula interdum sit amet nec lorem.

Figure 5.7 Disclaimer Chunk Rendition on Page

Marquee chunk template: This template (shown in Figure 5.9) should support an image or an animation object along with its attributes. The template should be designed by keeping future needs in mind. In the future, if we may want to use other assets in the marquee section (such as video or graphics), the template should be able to accommodate it. Hence, we have added a custom code section where we can accommodate such future needs through JavaScript. We can also add the Web Analytics JavaScript tag though custom code.

Right-hand section chunk template: Basically this template (shown in Figure 5.10) should be able to support content typically rendered in the right-hand page sections. Right-hand sections typically contain promotion content with assets (such as image, video) coupled with a brief description. Right-hand section chunk template provides attributes for adding images and video content and main content to author brief content. The template is flexible enough to add custom JavaScript code (through custom_code attribute), disclaimer (through disclaimer attribute), and footnote (through footnote attribute).

Product Category Chunk Template: This is the core content chunk template (shown in Figure 5.11), and the content model needs to support content for multiple columns. Hence, the template supports repeatable column content. The main sections in each of the columns provide support for authoring title, badge image, link, and main content (through body attribute). As with other templates, we can also specify disclaimer and footnotes.

Flex Section Chunk template: This template (shown in Figure 5.12) is a kind of "catch-all" that can be used to author content of varied structures. It provides a CDATA section within the body, and we can add any content (including free-form text/HTML).

Note: Though the template supports free-form text input, it is not recommended to add HTML or unstructured free-form text, as it defeats the purpose of structured content. The template can be used only in case of exceptional scenarios to create one-off content.

Content chunk management The main chunk management activities are:

- Reusability tracking: Construct a reusability matrix for content chunks. Reusability tracking table can be used to map individual chunks to pages and page section where it has been used.

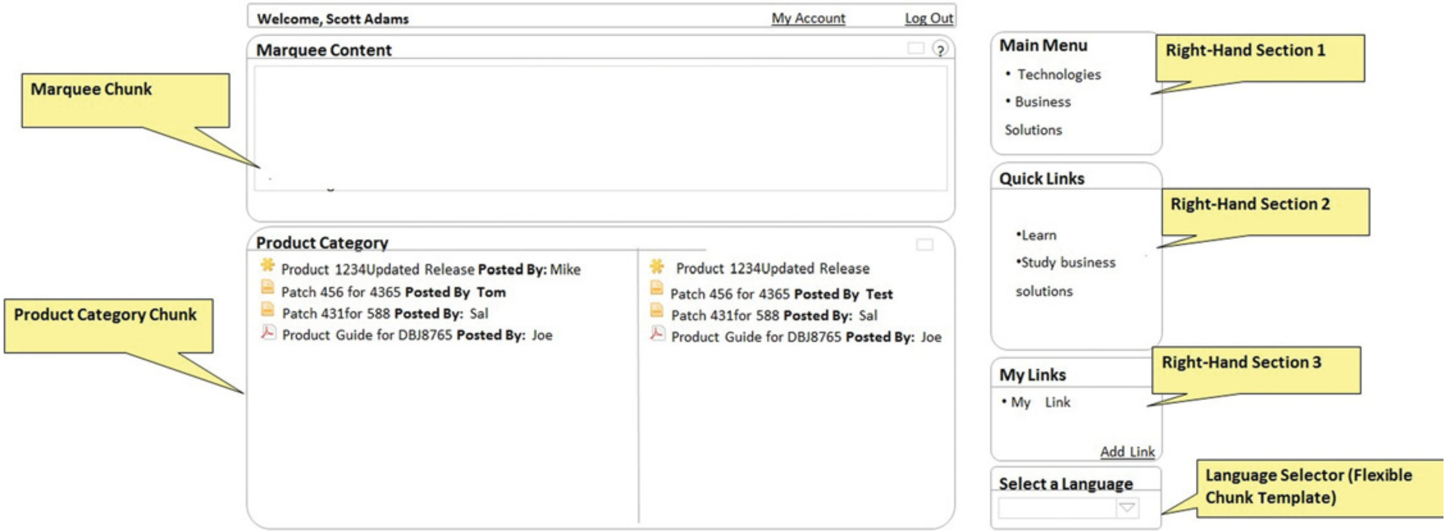
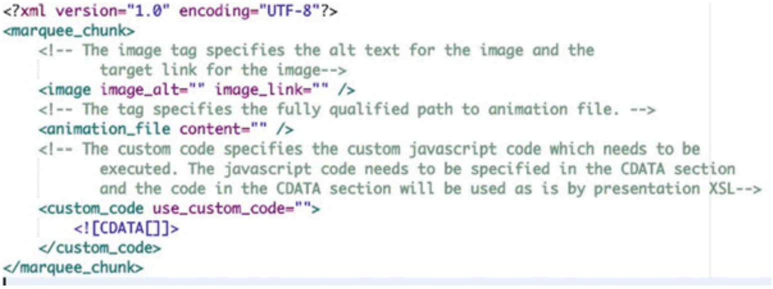

Welcome, Scott Adams
My Account
Log Out
Marquee Content
Marquee Chunk
Product Category
Product 1234Updated Release **Posted By:** Mike
Patch 456 for 4365 **Posted By Tom**
Patch 431for 588 **Posted By:** Sal
Product Guide for DBJ8765 **Posted By:** Joe
Product 1234Updated Release
Patch 456 for 4365 **Posted By Test**
Patch 431for 588 **Posted By:** Sal
Product Guide for DBJ8765 **Posted By:** Joe
Product Category Chunk
Main Menu
• Technologies
• Business Solutions
Right-Hand Section 1
Quick Links
•Learn
•Study business solutions
Right-Hand Section 2
My Links
• My Link
Add Link
Right-Hand Section 3
Select a Language
Language Selector (Flexible Chunk Template)

Figure 5.8 Product Category Page Consisting of Chunks

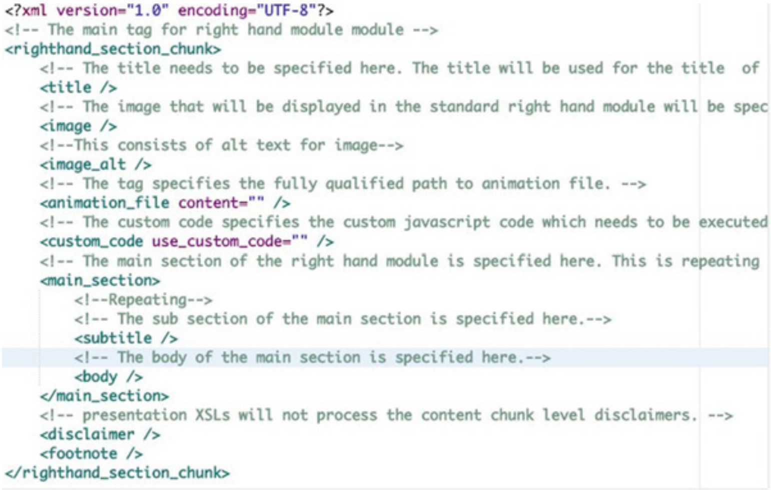


```
<?xml version="1.0" encoding="UTF-8"?>
<marquee_chunk>
    <!-- The image tag specifies the alt text for the image and the
          target link for the image-->
    <image image_alt="" image_link="" />
    <!-- The tag specifies the fully qualified path to animation file. -->
    <animation_file content="" />
    <!-- The custom code specifies the custom javascript code which needs to be
          executed. The javascript code needs to be specified in the CDATA section
          and the code in the CDATA section will be used as is by presentation XSL-->
    <custom_code use_custom_code="">
        <![CDATA[]]>
    </custom_code>
</marquee_chunk>
```

Figure 5.9 Marquee Chunk Template

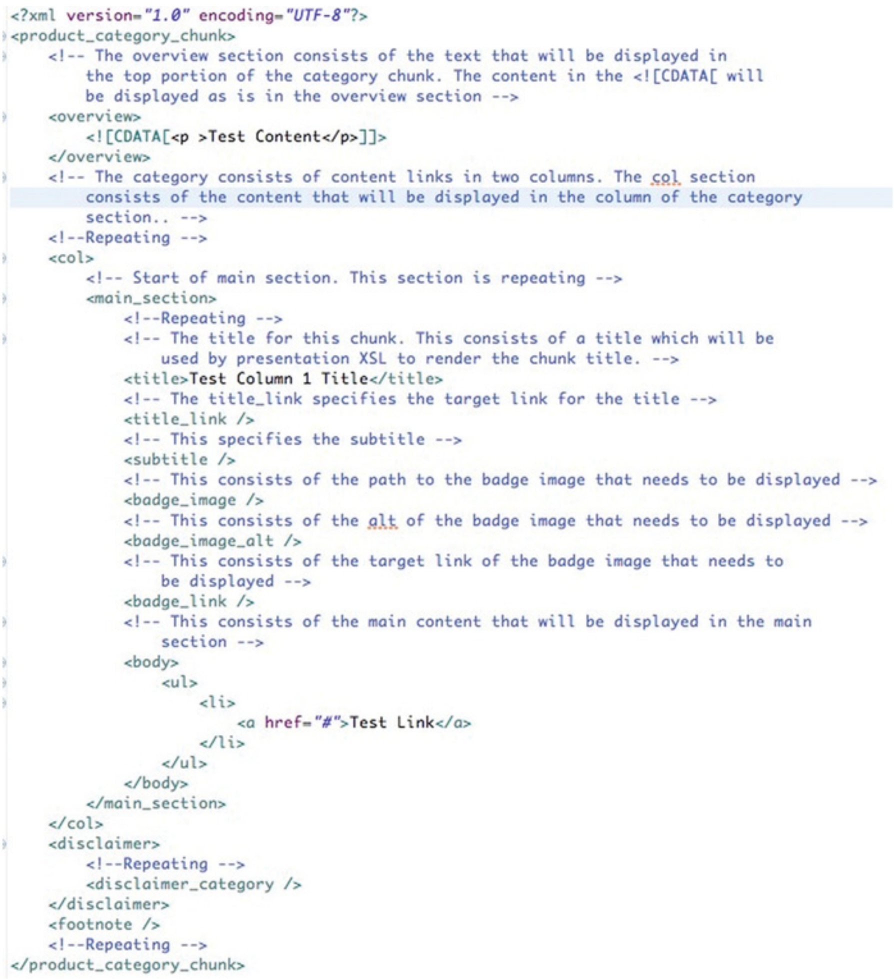
- Metadata selection: Develop robust metadata tagging process to tag appropriate metadata for the chunks to ensure that they are used in the relevant context.
- Automatic assembly: Create the process to use chunk metadata and assemble the larger content automatically.
- Chunk storage: For easier management, chunks should be named and logically categorized and stored in appropriate folders.

Chunk reusability governance We should ensure that the reusable content chunk is used for intended purposes and for target Web pages. For instance, when the author sets out to create the product details content for product2, then the author should be able to select the reusable product1 chunk (for example, the basic features could be same between product1 and product2, and hence we can reuse the product1's basic feature chunk) as applicable.



```
<?xml version="1.0" encoding="UTF-8"?>
<!-- The main tag for right hand module module -->
<righthand_section_chunk>
    <!-- The title needs to be specified here. The title will be used for the title  of
    <title />
    <!-- The image that will be displayed in the standard right hand module will be spec
    <image />
    <!--This consists of alt text for image-->
    <image_alt />
    <!-- The tag specifies the fully qualified path to animation file. -->
    <animation_file content="" />
    <!-- The custom code specifies the custom javascript code which needs to be executed
    <custom_code use_custom_code="" />
    <!-- The main section of the right hand module is specified here. This is repeating
    <main_section>
        <!--Repeating-->
        <!-- The sub section of the main section is specified here.-->
        <subtitle />
        <!-- The body of the main section is specified here.-->
        <body />
    </main_section>
    <!-- presentation XSLs will not process the content chunk level disclaimers. -->
    <disclaimer />
    <footnote />
</righthand_section_chunk>
```

Figure 5.10 Right-Hand Section Chunk Template

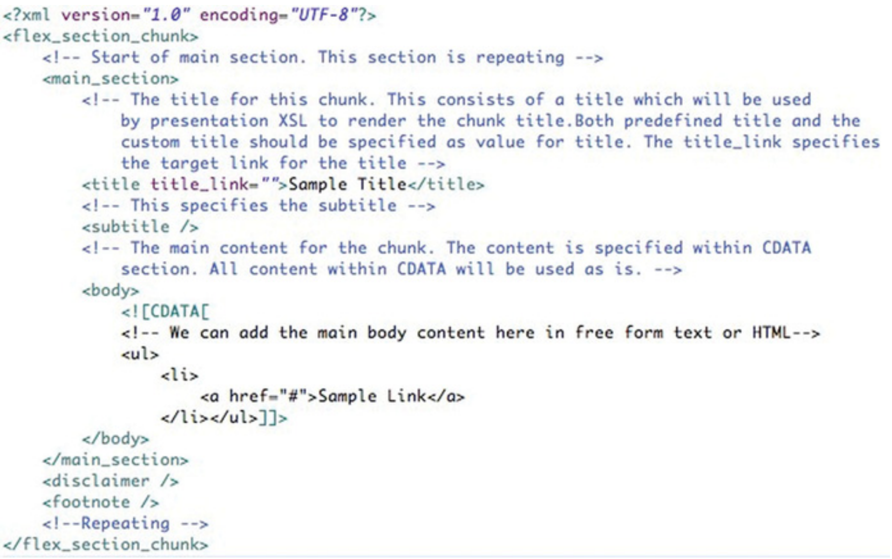


```
<?xml version="1.0" encoding="UTF-8"?>
<product_category_chunk>
    <!-- The overview section consists of the text that will be displayed in
        the top portion of the category chunk. The content in the <![CDATA[ will
        be displayed as is in the overview section -->
    <overview>
        <![CDATA[<p >Test Content</p>]]>
    </overview>
    <!-- The category consists of content links in two columns. The col section
        consists of the content that will be displayed in the column of the category
        section.. -->
    <!--Repeating -->
    <col>
        <!-- Start of main section. This section is repeating -->
        <main_section>
            <!--Repeating -->
            <!-- The title for this chunk. This consists of a title which will be
                used by presentation XSL to render the chunk title. -->
            <title>Test Column 1 Title</title>
            <!-- The title_link specifies the target link for the title -->
            <title_link />
            <!-- This specifies the subtitle -->
            <subtitle />
            <!-- This consists of the path to the badge image that needs to be displayed -->
            <badge_image />
            <!-- This consists of the alt of the badge image that needs to be displayed -->
            <badge_image_alt />
            <!-- This consists of the target link of the badge image that needs to
                be displayed -->
            <badge_link />
            <!-- This consists of the main content that will be displayed in the main
                section -->
            <body>
                <ul>
                    <li>
                        <a href="#">Test Link</a>
                    </li>
                </ul>
            </body>
        </main_section>
    </col>
    <disclaimer>
        <!--Repeating -->
        <disclaimer_category />
    </disclaimer>
    <footnote />
    <!--Repeating -->
</product_category_chunk>
```

Figure 5.11 Product Category Chunk Template

Reusability can be achieved mainly through two key aspects: *standard content structure* and *semantic metadata*. When the content structure is stored in a structured format (such as XML or DITA), it is easy to assemble content dynamically. Tagged metadata acts as a discovery tool. For instance, if both product1 and product2 are tagged with similar high-level metadata (say, *product family* = *adventure books*), then the CMS can be configured to provide the feature to list all the content chunks with matching product family during the authoring process.

Secondly, we need to effectively track the usage of a given content chunk across pages and sections. We can build automated reports to track this using the metadata (such as target_page or used_on_page). This reusability-tracking matrix helps in impact assessment during chunk updates and deletes.



```
<?xml version="1.0" encoding="UTF-8"?>
<flex_section_chunk>
    <!-- Start of main section. This section is repeating -->
    <main_section>
        <!-- The title for this chunk. This consists of a title which will be used
            by presentation XSL to render the chunk title.Both predefined title and the
            custom title should be specified as value for title. The title_link specifies
            the target link for the title -->
        <title title_link="">Sample Title</title>
        <!-- This specifies the subtitle -->
        <subtitle />
        <!-- The main content for the chunk. The content is specified within CDATA
            section. All content within CDATA will be used as is. -->
        <body>
            <![CDATA[
            <!-- We can add the main body content here in free form text or HTML-->
            <ul>
                <li>
                    <a href="#">Sample Link</a>
                </li></ul>]]>
        </body>
    </main_section>
    <disclaimer />
    <footnote />
    <!--Repeating -->
</flex_section_chunk>
```

Figure 5.12 Flex Section Chunk Template

A sample reusability-tracking matrix is shown in Table 5.2

This tracking metrics helps us map the usage of a content chunk on various pages and site sections. It may also help with impact analysis during changes to content chunks.

Template Guidelines and Best Practices

The following list provides some of the proven best practices while designing and using templates. It can be used as a checklist or as a reference guide for template design.

- Create the templates with reusability and extensibility as primary tenets. A template can be made reusable through flexible element structure (such as

Table 5.2 Reusability-Tracking Matrix

	Site section	Site pages	Web site
Product_short_ description chunk	Consumer section	Home page, Product home page	Main Web site
Product_features chunk	Consumer section, Reseller section	Product detail page, Product category page	Main Web site, campaign site, micro-site

repeatable and optional elements). Extensibility can be achieved through loose coupling and separation of concerns.

- Group the related entry fields into logical grouping of sections. For instance, a home page template can contain multiple repeatable sections, one each for content section on the page. Each section can contain elements such as text field, image selector, and rich text editor, which are required for a content section. This categorization helps the authors in easy authoring.
- Loose coupling and separation of concerns: The template should provide clear separation of content and presentation logic. For instance, a template can allow the author to add raw content, and the presentation logic can be added later through XSL files or presentation templates. This clear separation of concerns allows the template and content to be used across various devices. Similarly, the business logic should not be mixed with content or with presentation. Separating core content from its presentation is usually achieved through an authoring template (used for content creation) and a presentation template (for adding presentation markup to the content).
- Avoid styles and JavaScripts: Templates should avoid using in-line styles or JavaScript code while authoring content. As a general rule of thumb, any in-line presentation or formatting logic needs to be avoided while authoring content. This ensures that content is not tightly coupled with presentation logic (which in turn leads to tight coupling with the target device).
- Flexible security: It is always advisable to provide security controls at the template level and at the template-section level. This allows authors and reviewers to specify the intended target audience and user groups at a section level.
- Provide authoring aids: The template interface should provide aids such as validators, encoders, formatters, translators, and flexible editors that help authors and reviewers create quality content quickly.
- Create flexible template structure so that authors are able to add multiple sections and sub-sections and tag at sections level. Maintain consistent structure for templates across all languages.
- Preview functionality: Authors should be able to preview the overall look and feel of the page from the authoring interface. It is recommended to provide in-line editing option within templates to allow authors to preview content.
- DAM integration: Provide author-friendly components to attach the assets through a DAM system. This includes importing assets from DAM, drag-and-drop assets to template, etc.
- Tagging and metadata: Templates should provide elements to attach the relevant metadata and tags to content. Authors should be able to tag at a page level as well as at a section level.
- Friendly content categorization: The content template should allow the author to save authored content to a meaningful category.

- Create custom components with reusability as a primary goal. Some examples of reusable customer components include page header and footer, search component, navigation components, carousel container, widget container, etc.
- Provide search capability within template so that authors can search relevant content chunks, images, and files without leaving the authoring screen.
- Templates should allow authors to specify SEO and Web analytics tags and scripts wherever necessary. Authors can use this feature to add page-wide SEO and analytics scripts.
- The authoring and presentation templates should support UTF-8 character encoding to ensure that content in various character sets is supported.
- If the templates allow editing or authoring XML content, then the entered content should be validated using XSD (XML Schema Definition).
- For each of the template fields and sections, provide context-sensitive help, which may explain the way the corresponding template fields are intended to be used.
- Wherever possible, use descriptive names/labels for the template fields to avoid conflicts and confusion. For instance, instead of labeling "Description," use "Product Description" to label the product description input field.
- Provide built-in support for a DAM system (for image/asset selector fields), a workflow system (for specifying and adding content to the workflow), and a metadata system (for tagging content with metadata).
- If the solution needs a large number of templates, create a logical grouping of templates. The grouping can be based on site sections, content models, or language/geo. Template naming conventions should be uniform and consistent. Authors should be able to easily select the appropriate template using groups and naming conventions.

Note: We have given sample template checklist in Appendix B, which can be used for template design.

5.4 TEMPLATE SUPPORT AMONG VARIOUS CMS

All CMS platforms support content templates as part of their core feature. Drupal has built-in templates,and we can also leverage the content template (ConTemplate) module in addition to this. CKEditor and WYSIWYG editors provide direct-to-render content in Drupal. Drupal supports creation of structured content through other features: Field API for modeling granular content; Filters to add presentation for core content; and View modes and views module for reusing content across various presentation formats. The REST module exposes content to external systems in JSON, XML, and other formats.

WordPress supports templates that govern the site rendition. Templates use data from a database and render the HTML output. Template files generate header and footer and invoke other templates to render core blog content on the Web site.

Template tags can be used to add dynamic information to the blog post. Joomla templates mainly provide the overall design and look and feel of the site. ECM platforms such as Documentum, Interwoven TeamSite, and Alfresco support the robust template framework.

5.5 CASE STUDY: BUILDING CONTENT TEMPLATES FOR A WEB SUPPORT SITE

In this section we take a look at the detailed steps of how we can design content templates for a Web site. This case study assumes that we are building the site from ground up.

Web Support Site Case Study: Context and Overview

An organization wants to build a robust online support platform for its products based on a self-service model. The main business drivers are to enable the customers to discover the information quickly and to provide them with decision tools to get the support so as to reduce the support call volume. Online support is mainly provided through solution knowledge base, product information pages, and real-time collaboration tools such as video chat, wiki, support forums, communities, and incident system. The organization has selected a CMS platform after due diligence and now wants to build the self-service platform using the CMS.

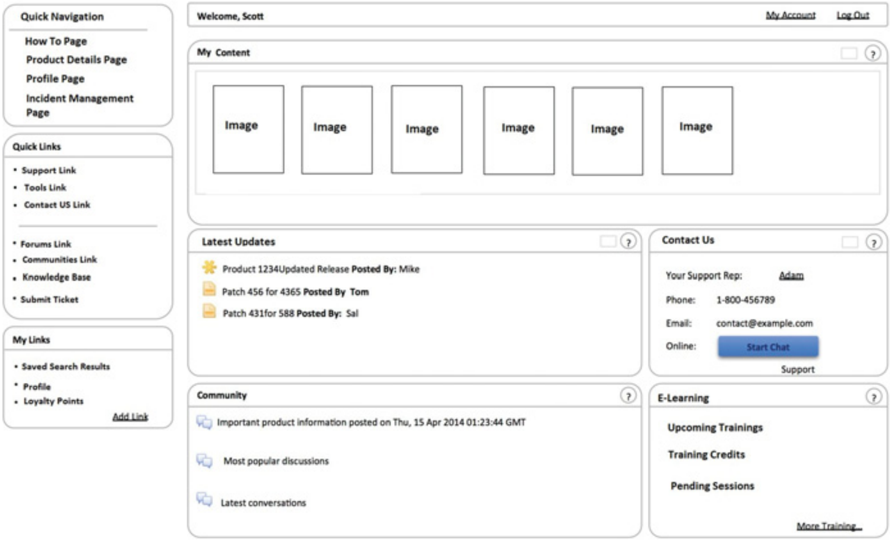
Step 1: Creation of wireframes and mockups

The organization engages a creative agency to design the user experience for the site. The agency creates the wireframes (page blueprints) for all the pages. The site hierarchy has mainly two levels:

- First-level pages: Login page (for user authentication), home page (displaying campaigns and key support related information), landing page (support information based on customer type), and search page (for solution search)
- Second-level pages: Solution articles pages (to display detailed solution article retrieved from knowledge base), product detail pages (to provide detailed product content), incident page (for logging and tracking incidents), wiki page (for creating solution articles), forum page (for discussing solutions and products), and community page (for collaboration among like-minded individuals)

The site hierarchy is also reflected in the menu, left navigation, and other navigation elements.

A sample wireframe of the home page and landing page is shown in Figure 5.13, and a wireframe for the landing page is depicted in Figure 5.14



Quick Navigation

How To Page
Product Details Page
Profile Page
Incident Management Page

Quick Links

- Support Link
- Tools Link
- Contact US Link
- Forums Link
- Communities Link
- Knowledge Base
- Submit Ticket

My Links

- Saved Search Results
- Profile
- Loyalty Points

Add Link

Welcome, Scott My Account Log Out

My Content ?

Image Image Image Image Image Image

Latest Updates ?

Product 1234Updated Release **Posted By:** Mike
Patch 456 for 4365 **Posted By** Tom
Patch 431for 588 **Posted By:** Sal

Contact Us ?

Your Support Rep: Adam
Phone: 1-800-456789
Email: contact@example.com
Online: Start Chat
Support

Community ?

Important product information posted on Thu, 15 Apr 2014 01:23:44 GMT
Most popular discussions
Latest conversations

E-Learning ?

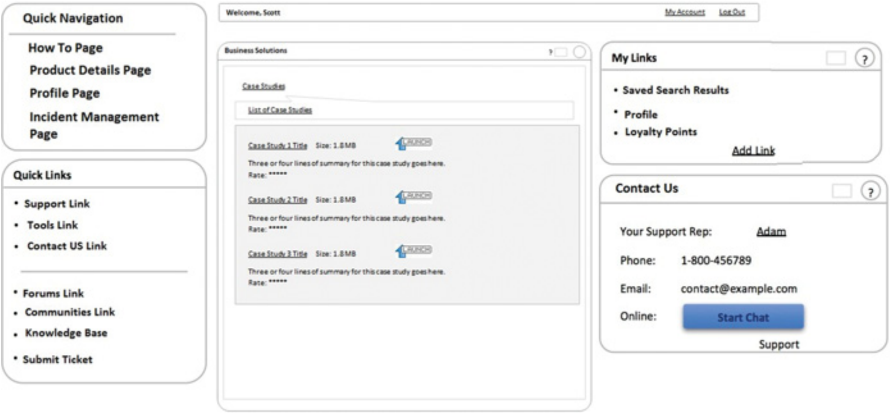
Upcoming Trainings
Training Credits
Pending Sessions
More Training...

Figure 5.13 Home Page Wireframe

Step 2: Creation of content master templates

As discussed earlier, we need to identify the unique content structures for the site pages before we begin creating the master templates. Master templates will be designed so that they can be easily extended and reused for multiple page content structures. Usually the pages at the same level tend to have uniform/consistent content structure to ensure consistent user experience. Analyzing the content structure of first-level pages and second-level pages is the first step in this direction.

Note: One of the principal best practices in designing the content template is to separate content from its presentation. Hence, in this case study our primary focus is



Quick Navigation

How To Page
Product Details Page
Profile Page
Incident Management Page

Quick Links

- Support Link
- Tools Link
- Contact US Link
- Forums Link
- Communities Link
- Knowledge Base
- Submit Ticket

Welcome, Scott My Account Log Out

Business Solutions ?

Case Studies

List of Case Studies

Case Study 1 Title Size: 1.8MB LAUNCH
Three or four lines of summary for this case study goes here.
Rate: *****

Case Study 2 Title Size: 1.8MB LAUNCH
Three or four lines of summary for this case study goes here.
Rate: *****

Case Study 3 Title Size: 1.8MB LAUNCH
Three or four lines of summary for this case study goes here.
Rate: *****

My Links ?

- Saved Search Results
- Profile
- Loyalty Points

Add Link

Contact Us ?

Your Support Rep: Adam
Phone: 1-800-456789
Email: contact@example.com
Online: Start Chat
Support

Figure 5.14 Landing Page Wireframe

mainly on designing the templates to cater to the content model needed for the site. We assume that presentation logic (such as content formatting, styling, etc.) is injected into core content through XSL transformation or through presentation templates.

Upon analysis we can find that we can create master templates for content on the following set of pages (with each set of pages forming a logical group):

- Home page and landing page
- Login page and search page
- Solution article pages and product detail pages
- Forum pages and community pages

All pages in a given group have similar content structure (with slight variations in content blocks). Variations can be addressed by using flexible features in a template, such as optional elements, repeatable elements, and conditional content section rendering. So we create one master template for each of the above groups. All pages belonging to that group can extend or reuse the master template during authoring. For example, "My Content" of the home page contains repeatable images with titles laid out. The "Business Solutions" section of the landing page too has a similar structure (with title and brief description) laid out vertically. The differences between two content chunks are image and brief description elements. We can easily accommodate for this variation using the repeatable sections, placeholder sections, and the optional images feature in the master template. Hence, both home page and landing pages (for each of the customer types) can be constructed using the same master template

Step 3: Creation of all content templates

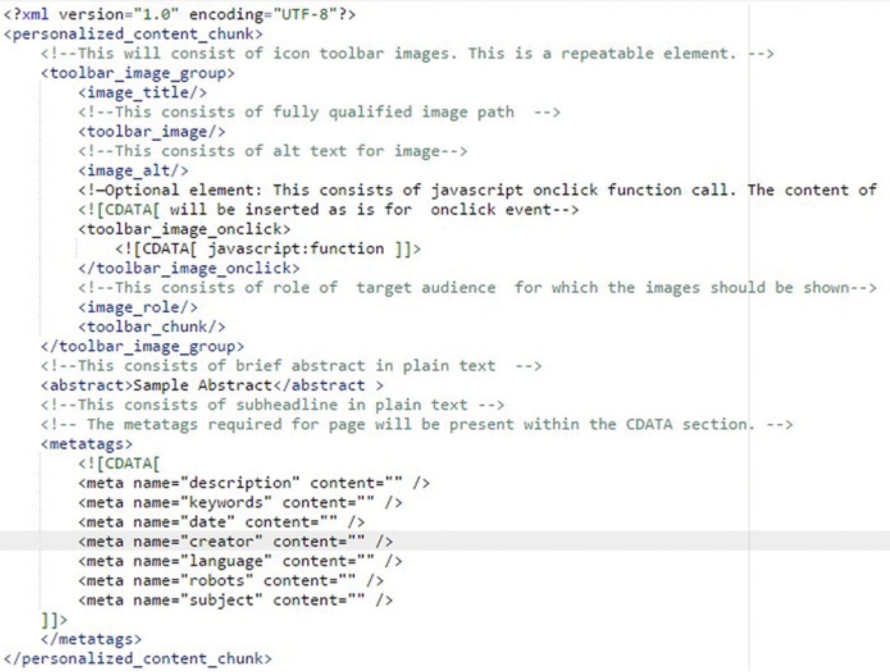
We can identify the structure of the rest of the pages and design templates for corresponding content models. In this scenario, wiki pages, forum pages, community pages, and incident pages are the ones that are remaining. The content for forums, wiki, and community pages are mainly created by end-users through user-generated content (UGC). So there would not be content-authoring requirement for these pages. For the incident page we can create a template that renders an input form for the end-user to fill in the incident details.

In this case study we are using XML-based content templates.

Note: If we plan to use the CMS-provided template features, we can follow the same steps. XML template elements would become the CMS template fields.

A content template (in XML format) for the "My content" chunk of home page is shown in Figure 5.15. The template offers a provision to input multiple repeatable images and other required elements such as image title, section abstract, and chunk keywords. The "image_role" is used by the presentation engine (such as portal or XSL transformation engine) to filter the images based on the role of the logged-in user.

A user interface can be created based on this XML and enforce the policies/restrictions using XSD. Alternatively we can use template-user interfaces to achieve similar functionality. In such a case, XML elements will translate into fields in the template.



```
<?xml version="1.0" encoding="UTF-8"?>
<personalized_content_chunk>
    <!--This will consist of icon toolbar images. This is a repeatable element. -->
    <toolbar_image_group>
        <image_title/>
        <!--This consists of fully qualified image path  -->
        <toolbar_image/>
        <!--This consists of alt text for image-->
        <image_alt/>
        <!–Optional element: This consists of javascript onclick function call. The content of
        <![CDATA[ will be inserted as is for  onclick event-->
        <toolbar_image_onclick>
            <![CDATA[ javascript:function ]]>
        </toolbar_image_onclick>
        <!--This consists of role of  target audience  for which the images should be shown-->
        <image_role/>
        <toolbar_chunk/>
    </toolbar_image_group>
    <!--This consists of brief abstract in plain text  -->
    <abstract>Sample Abstract</abstract >
    <!--This consists of subheadline in plain text -->
    <!-- The metatags required for page will be present within the CDATA section. -->
    <metatags>
        <![CDATA[
        <meta name="description" content="" />
        <meta name="keywords" content="" />
        <meta name="date" content="" />
        <meta name="creator" content="" />
        <meta name="language" content="" />
        <meta name="robots" content="" />
        <meta name="subject" content="" />
    ]]>
    </metatags>
</personalized_content_chunk>
```

Figure 5.15 My Content Chunk Template

Step 4: Adding workflow, SEO, analytics, content category, metadata, and other content elements

The content template should also support fields for authors to add SEO tags, analytics tags, metadata values, and content category and assign the workflow for the content. These supporting fields are required for optimal content discovery and for content search.

For the functioning of the Web site we also need other elements such as navigation elements (left navigation, breadcrumb, menus, etc.), static content fragments (such as about us pages, footer page), dynamic content fragments (such as Web service or feed-based content), and so forth. We can create most of these as CMS components (a custom widget, an include file, or a dynamic component). We can also use one of the content templates for authoring static content fragments. For dynamic content blocks we can provide a flexible dynamic template wherein the author can add the necessary JavaScript (or equivalent dynamic code) along with placeholder elements. Flexible dynamic templates are normally used for construction of widgets, dynamic navigation elements, and other integration components to retrieve information from Web services and feeds.

Note: The templates can be used to author the entire page or individual page sections. If the page sections are reused across multiple pages (such as header/footer, left navigation, right hand boxes, etc.), we can create templates to create such sections *through content chunks. A page presentation engine can assemble all the individual page sections to render the complete page.*

Step 5: Mapping of Web page sections to template sections

Here is how these page sections can be constructed by using templates and otherwise.

- Header: This is the most prominent section of the page, which contains first-level menus (with an expanding feature to second-level menus), organization logo, global search, welcome link, and similar elements. As header is a reused across various pages, we can use a separate template for creating it. The header template can contain the combination of dynamic module and regular static fields. As header menus are dynamic (they change if the site structure changes, and are based on user access levels), we can use a flexible dynamic template to construct this menu. The JavaScript code added to the dynamic template section can read the menu structure from a JSON/XML and can construct the menu dynamically. The remaining sections of the header are static, and hence we can use any static template to author this content. Usually the header section also contains the master CSS includes, web analytics scripts, and similar global elements. Again, these can be added along with static content using the static template section of the header template.
- Left navigation: This is the first column in the page layout. Like menus, left navigation is dynamic, and we can use a flexible dynamic template for this.
- Main page sections: As we can see, there are repeatable content sections/modules in the second column. The content sections can be a combination of static and dynamic content. We can use the appropriate fields in the master template to author this. The visibility rules can be configured to the repeatable sections in the master template.
- Right-hand boxes: Content modules in the third column are mostly modules such as “Quick links,” “Search box,” “Saved search,” and the like. As these are reused across multiple pages, we can again use generic templates to author content for these modules.
- Footer section: As this section is mostly static, we can use a flexible content template to author this reusable content section.
- Navigation bars: Optionally we can have a horizontal navigation bar, which provides the links to popular sections and pages. This again can be developed using a flexible dynamic template to populate the links.

5.6 CONTENT WORKFLOWS

Workflows are an integral part of the content management system. They help model complex business processes through a sequence of steps, with each step handling its own set of business rules and having its own security restrictions. Workflows are mainly used for content authoring, reviewing, editing, publishing, translation, and other similar activities. Content workflows mainly define the following:

- Ordering and sequencing content activities of a business process
- Enforcing business rules and implementing business processes and policies
- Enforcing security policies for performing each of the tasks through security roles
- Information flow across tasks
- Notification and reporting of status of tasks

A sample authoring and review workflow in Drupal CMS is shown in Figure 5.16

In the Drupal authoring flow, authors add content and metadata to the content node. Once content is complete, it triggers the approval workflow. The approver can review submitted content through the workflow inbox and can either approve or reject. If approved, content is sent for publishing; if rejected, it is sent back to the author. In the publishing process, live content is unpublished and then new content is published.

Workflow steps and actions are restricted based on user roles. A user can take action (such as review, approve, reject), after which the workflow moves to the next stage. At each step workflow also notifies the configured admins.

Workflow Design

Almost all important content management tasks (such as authoring, publishing, translation, etc.) are implemented through workflows. CMS should support the following workflow features:

- Modeling business processes: We should be able to model the workflow based on the underlying business processes.
- Workflows should be able to execute activities in a predefined sequence with appropriate business rules (such as routing rules, branching conditions, etc.).
- Assigning users and roles to workflow steps and actions: This provides role-based access control for each of the workflow steps.
- E-mail notification: CMS should be able to add notification features to update workflow status to the key stakeholders upon completion of each workflow step.
- Support for complex processing: We should be able to configure complex workflow features such as parallel workflow steps, decision-making steps, sub-workflows, pause/restart workflow, time-bound workflow steps, automated advancing, service integration, etc.
- The workflow administrator should be able to impersonate and take actions on behalf of other users to handle any workflow exceptions.

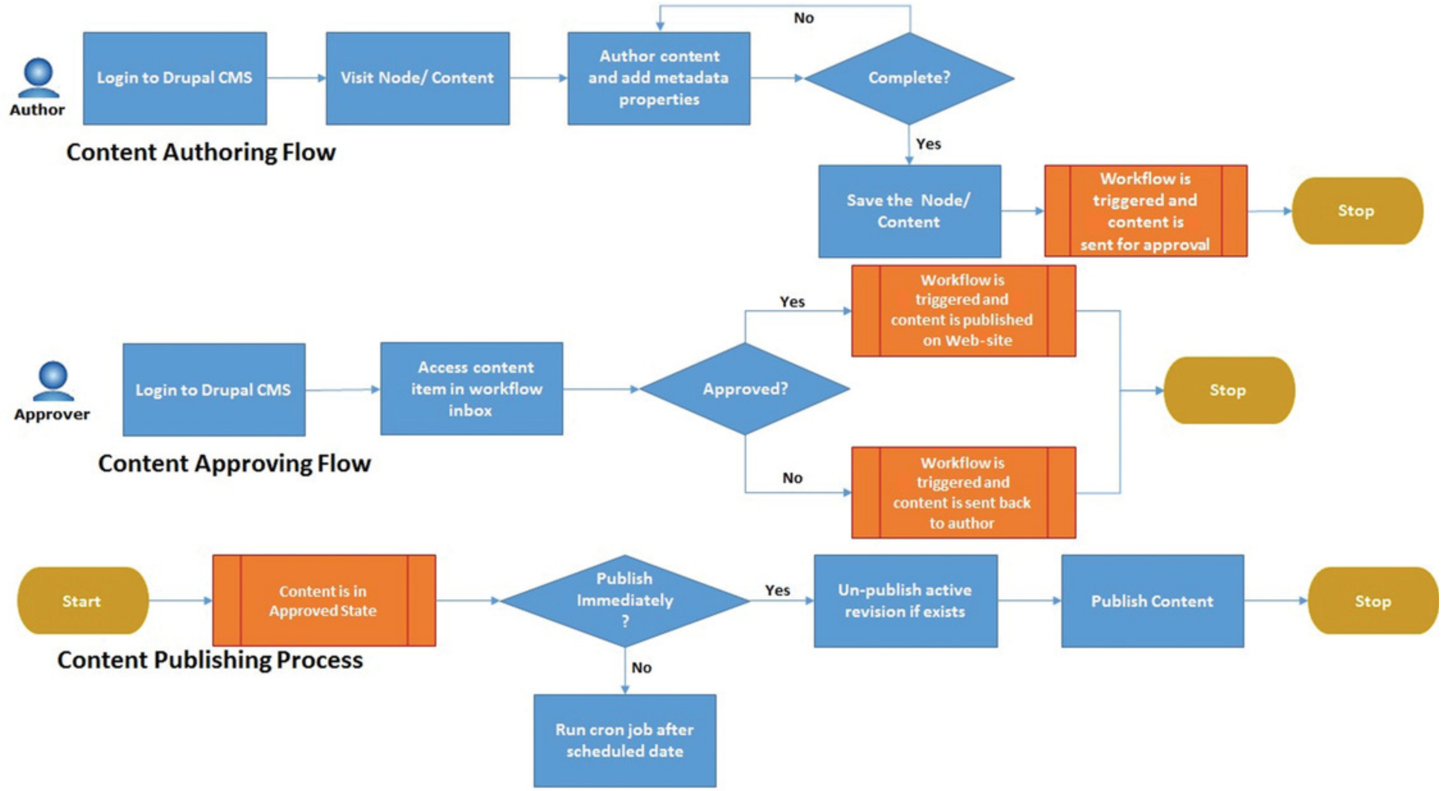

Author
Login to Drupal CMS
Visit Node/ Content
Author content and add metadata properties
No
Complete?
Yes
Content Authoring Flow
Save the Node/ Content
Workflow is triggered and content is sent for approval
Stop
Yes
Workflow is triggered and content is published on Web-site
Approver
Login to Drupal CMS
Access content item in workflow inbox
Approved?
Stop
Content Approving Flow
No
Workflow is triggered and content is sent back to author
Start
Content is in Approved State
Publish Immediately ?
Yes
Un-publish active revision if exists
Publish Content
Stop
Content Publishing Process
No
Run cron job after scheduled date

Figure 5.16 Sample Content Workflows

- The workflow component should be able to handle the deadlock scenario through appropriate notifications.

Prerequisites for the workflow design

Here is a look at the prerequisites for designing the content workflow:

- **Defining content strategy:** Content strategy provides basic understanding of content goals, content processes, content governance, content planning, and the like. Once the content strategy is in place, workflows can use the defined content governance structure and other content best practices (such as content publishing formats, content standards, etc.).
- **User interviews:** We need to interview all the business stakeholders and system users (such as authors, editors, reviewers, approvers, marketing team, etc.) to understand the needs, gaps, and other requirements. This will help us understand the current content processes and associated business rules. This will be done as part of CMS discovery and planning phase of CMS implementation approach.
- **Definition and mapping of content roles:** All appropriate roles (such as author, reviewer, and approver) should be defined and mapped to process steps and responsibilities.
- **Identification of workflow task items:** Compile the list of all tasks and business rules that are involved in a workflow, such as:
  - Detailed steps in a content approval process
  - List of all process prerequisites and dependencies
  - List of approvals needed for content publishing
  - Process to attach images of different resolution for Web pages
  - Process for content translation
  - Process to add SEO, analytics, and metadata to content
  - Business rules for content deletion and archival.
- **Exception handling:** We should also list the “What-if” scenarios that may arise in the content processes, such as:
  - What if the approver is not available?
  - What is the maximum time the process step can wait for review?
  - What is the process for emergency and fast publishing?
  - How should the exception be handled if the workflow step errors out?
  - What is the fallback mechanism if the translation service is down?
  - How do we handle a long-running workflow step?
- **Compile all the best practices and checklists:** Workflow users can use these best practices and checklists as job aids to ensure that tasks are executed consistently. For instance, the editorial and visual guidelines for content authors and image specifications for content chunks will be handy for completing the tasks.

- **Compile all the tools and accelerators:** We can also compile all tools, standards, and accelerators that will be useful while working on workflow steps. W3C standards, accessibility standards, and HTML standards are some of the examples in this category.
- **Addressing existing gaps:** We can use the workflow as an opportunity to address current process gaps. Compile all current gaps and challenges (such as too many steps in publishing workflow, lot of manual steps in review process, etc.).
- **Integrations:** All necessary integrations should be done before modeling workflow. Integration with DAM system, metadata system, translation system, and taxonomy system are essential in completing the workflow steps, and hence they need to be completed and verified before designing workflows.

Designing Workflow

As mentioned earlier, workflows are closely modeled on the underlying business processes. The key steps in designing a content workflow are:

1. **Current process analysis:** Closely look at the current business processes related to content. This includes content creation, content review, content approval, content publishing, etc. Some of the common scenarios that need workflows are: publishing workflow for publishing content to various publishing targets; translation workflow for content translation; content approval workflow for authoring, editing, reviewing, previewing, and approving the content; and asset rendition workflow wherein a document is transformed into various renditions or an image is transformed into various resolutions.
2. **Business process modeling:** One of the activities in modeling a business process is to document and analyze the involved business rules. The business rules normally cover scenarios such as skill-based routing of workflow steps, any location or time zone–based rules, validation requirements, notification requirements, exception handling, fallback logic to handle unavailability of reviewers, time-bound workflow steps, etc. For instance, a content-authoring and publishing flow would contain a number of business rules such as:
   - **a.** Role-based access to authoring, editing, and publishing functions
   - **b.** Rules to select the publishing location based on language
   - **c.** Levels of approvals required before final publishing
   - **d.** Fallback rules if approvers are not available
   - **e.** Routing rules based on the reviewer's skillset
   - **f.** Rules to manage simultaneous content updates

   These business processes and business rules should be modeled in a diagram (such as process flow diagram or flow diagram). They can then be implemented in CMS using its built-in workflow tool. For each of the workflows, the goals for monitoring should be defined.

3. **Role assignment:** Assign the roles for each of the content workflow steps and set the permissions (such as approve, reject) accordingly.
4. **Integration:** Integrate the workflow steps to external/internal systems and services needed to complete the task. For instance, a translation step needs to invoke translation Web service and hence needs to be integrated with the translation management system (TMS).
5. **Optimization:** Optimize the workflow by using techniques such as parallel execution, auto-approvals, time-bound approvals, one-step publishing, etc. Ensure that the workflow can handle emergency, one-off content publishing mechanism to respond to emergency needs.
6. **Testing:** Test the workflow for all exception scenarios compiled during initial analysis. Also test the workflow at various content loads and in various locales to ensure that it works as expected.
7. **Monitoring:** Monitor the adoption of the new workflow and actively seek feedback from its users. Track the workflow effectiveness against the predefined metrics. Monitoring is used to continuously improve the workflow process.

In a single-step workflow we have a single approver for content created, and in a case of multi-step workflow, we have two approvers who sign off on content. In some complex enterprise scenarios, various groups have a stake in content; for instance, marketing team and/or legal team may need to review content with associated business rules (such as auto-approval rules, fallback approval rules, etc.).

Content metadata during workflow stages When content moves from one workflow step to another, workflow engine and other CMS components can track it through its metadata. Usually the workflow updates the internal metadata (such as “content stage” or “workflow stage”) when content progresses through workflow steps. Some of the sample values of workflow metadata are “Draft,” “Submitted,” “First-level approved,” “Second-level approved,” “Legal approved,” “Published,” “Expired,” etc. The workflow engine will also send e-mail notifications when content moves through each stage.

Workflow Optimization

The main techniques we can use to optimize the workflows are:

- Identify the duplicate and unnecessary steps in workflow and eliminate them.
- Explore opportunities to convert sequential steps to execute in parallel. The steps that have no mutual dependency are ideal candidates for parallel execution.

- Identify the workflow and process steps that are time and effort consuming. These steps need to be further analyzed for optimizations. We can look at opportunities to automate such activities or to execute them in parallel.
- If any of the workflow steps involve long-running synchronous transaction (thereby having long wait times), we need to convert them to an asynchronous model with the help of callback functions. For instance, in a translation workflow, the final step would invoke the translation service of the translation management system. Since the translation usually takes a long time, we can convert the synchronous call to translation service to an asynchronous call and provide the callback service name to the TMS (pausing the workflow). Once the translation is complete, TMS would invoke the callback service to notify the status and restart the workflow.

For time-sensitive steps, we can implement time-bound auto-forward features. For instance, if any of the approval steps take longer than business timeout value, that step can be made auto-approved, and the workflow will forward to next step automatically.

Workflow support in CMS platforms

Drupal supports a feature-rich workflow through the "Workflow" module. We can assign a security role for each workflow state, and actions can be triggered when the workflow transitions from one state to another. In WordPress we can use the Oasis Workflow plugin module. The plugin provides visual workflow designer, role-base routing, notifications, and other features. We can leverage extensions such as "My Content & Workflow" to implement workflows in Joomla! Liferay uses Kaleo workflow engine for modeling process steps.

5.7 CASE STUDY: MODELING WORKFLOW FOR A KNOWLEDGE MANAGEMENT SYSTEM

In this section we look at the stages in workflow creation through a case study. We apply the steps we discussed earlier to design the workflow. We assume that all prerequisites are satisfied.

Let us look at a workflow design for authoring artifacts for the knowledge management (KM) system. Normally KM systems need solution articles (such as How-to articles, Troubleshooting tips, etc.), and they need to be reviewed by all concerned stakeholders to ensure accuracy and completeness of content.

For this case study, there are three levels of review. The first step in designing a workflow is to understand the business process and accompanying business rules and security roles. Table 5.3 summarizes the workflow roles, authoring process, and business rules for knowledge article publishing workflow. The steps are given in chronological order.

The next step is to model the business process to satisfy all the business rules. The business process diagram is shown in Figure 5.17

Table 5.3 Workflow roles, activities, and rules in publishing workflow

Workflow Role	Workflow Stage	Process Activities and Business Rules
Creator	Initiation Stage	• Creates the knowledge article draft using the template • Can review the draft article • Can initiate image request workflow • Post-completion, the workflow moves to the review phase (which changes the workflow status to “ready for review”)
Reviewer	Initial Review Stage	• Will get an e-mail notification once an article is “ready for review” • Can review and edit the initial draft of the knowledge article • Can review the final draft of the knowledge article • Post-completion, the workflow moves to the group review phase (which changes the workflow status to “approved by reviewer”)
Group Reviewer	Group Review Stage	• Will get an e-mail notification once an article is “approved by reviewer” • Provide additional review comments after the initial review stage • Article will be auto-approved based on article priority (P1 article in 2 hours, P2 article in 10 hours, and P3 article within 2 days) • Post-completion, the workflow moves to the marketing review phase (which changes the workflow status to “approved by group reviewer”)
Marketing Reviewer	Marketing Review Stage	• Will get an e-mail notification once an article is “approved by group reviewer” • Provide marketing review comments • Can approve or reject the workflow. If rejected, the workflow will go back to the Initiation stage. • Post-approval, the workflow moves to the legal review phase (which changes the workflow status to “approved by marketing reviewer”)
Legal Reviewer	Legal Review Stage	• Will get an email notification once an article is “approved by marketing reviewer” • Provide legal review comments • Can approve or reject the workflow. If rejected, the workflow will go back to the Initiation stage. • Post-approval, the workflow moves to the final review phase (which changes the workflow status to “approved by legal reviewer”)

Table 5.3 (*Continued*)

Workflow Role	Workflow Stage	Process Activities and Business Rules
Final Reviewer	Final Review Stage	• Will get an -mail notification once an article is "approved by legal reviewer" • Performs final review • Can approve or reject the workflow. If rejected, the workflow will go back to the Initiation stage • Once approved, the article will be published along with all associated assets • A new article version will be created post-publishing. • Can initiate localization workflow

Figure 5.17 depicts roles and the workflow activities for each role. The workflow creator can initiate and assign the workflow to a skilled writer. The workflow writer accepts the workflow and creates/edits the content. The writer may also initiate the image request flow if content needs images. Post-authoring, content is sent to the

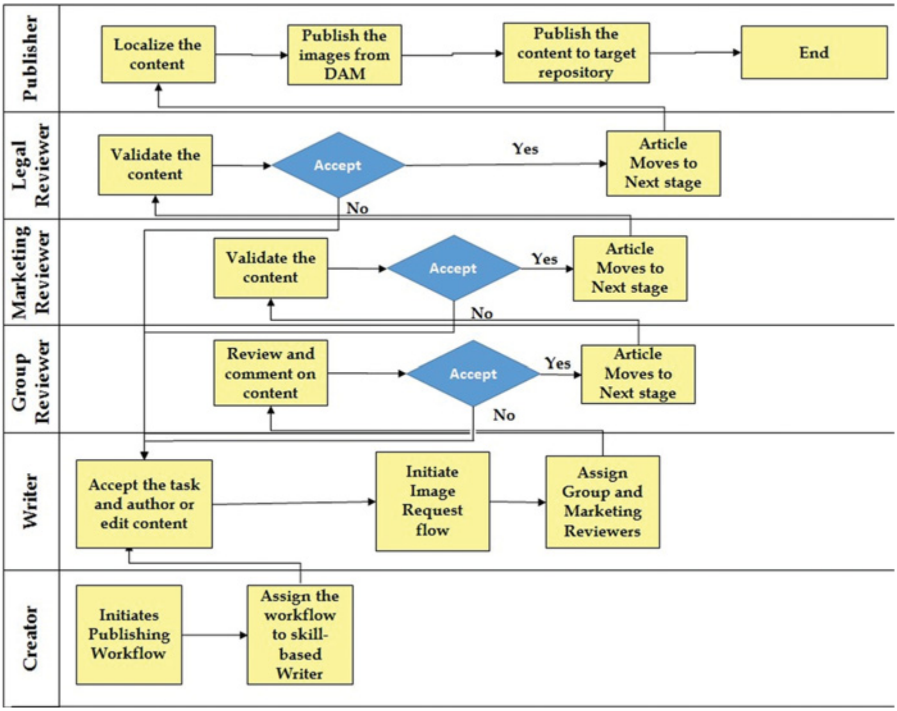

Publisher
Localize the content
Publish the images from DAM
Publish the content to target repository
End
Legal Reviewer
Validate the content
Accept
Yes
No
Article Moves to Next stage
Marketing Reviewer
Validate the content
Accept
Yes
No
Article Moves to Next stage
Group Reviewer
Review and comment on content
Accept
Yes
No
Article Moves to Next stage
Writer
Accept the task and author or edit content
Initiate Image Request flow
Assign Group and Marketing Reviewers
Creator
Initiates Publishing Workflow
Assign the workflow to skill-based Writer

Figure 5.17 Business Process Diagram for Knowledge Article Publishing

group reviewer, marketing reviewer, and legal reviewer for approval, in that order. Each of the approvers can validate and accept content, which moves it to the next stage; if any of the approvers rejects content, it comes back to the writer for rework and updates. After all the approvals, the publisher localizes the content (through translation workflow) and publishes it along with the associated images from DAM to target repository.

Once the business process is modeled, we can implement the business process using workflow tools supported by the corresponding CMS.

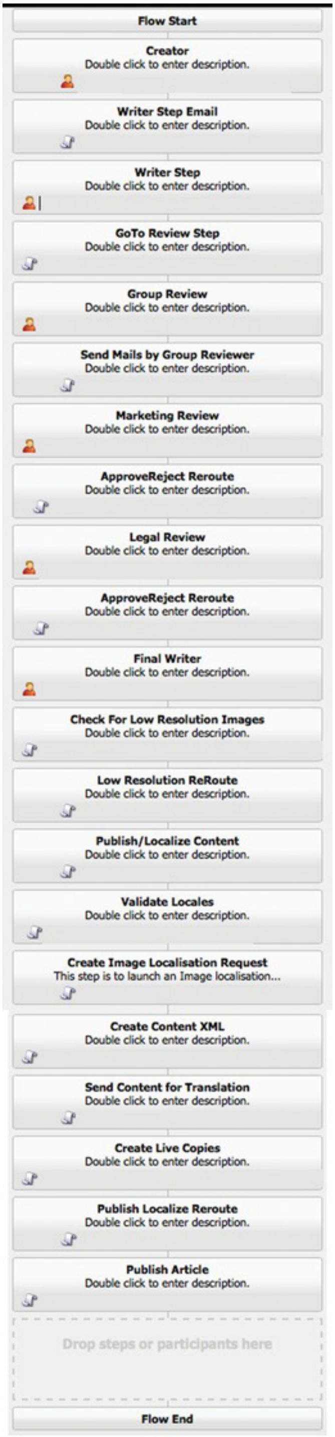
A sample implementation of above business process is implemented in the CMS as shown in Figure 5.18

Note: We have modeled the business process in Figure 5.18 using Adobe AEM workflow feature.

Some of the CMS workflows provide decision rules, split steps, and other features to implement the business process. In this case, each of the workflow steps is processed by underlying Java Classes, which also handle the business rules and related conditions.

5.8 CHAPTER SUMMARY

- Templates are reusable components for creating structured content.
- Authoring templates are components for authoring the content for Web pages.
- Presentation templates add the required style elements to the core content.
- The main design considerations for authoring template are template reusability, flexibility, support for granular content, specification of metadata, SEO, analytics, and workflow.
- Prerequisites for template design are compiling inventory of page layouts and content model, content model attributes, data types, and integration with DAM, MMS, TMS, and required systems.
- Main steps in template creation are template elements creation, adding workflows, adding presentation template, and assigning permissions.
- Presentation templates add presentation formatting and UI elements to the core content.
- Some of the main template best practices are designing templates for reusability, extensibility, flexibility, and loose coupling. Templates should also provide authoring aides, fields for tags, SEO, Web analytics, support UTF-8 encoding, and provide descriptive element names.
- The chapter discussed a detailed case study for a Web support site. The steps involved are wireframe creation, master template creation, template creation, and finally adding SEO, analytics, category, and metadata to the template.
- Workflows model the business process and provide sequential activity execution.



Flow Start

Creator
Double click to enter description.

Writer Step Email
Double click to enter description.

Writer Step
Double click to enter description.

GoTo Review Step
Double click to enter description.

Group Review
Double click to enter description.

Send Mails by Group Reviewer
Double click to enter description.

Marketing Review
Double click to enter description.

ApproveReject Reroute
Double click to enter description.

Legal Review
Double click to enter description.

ApproveReject Reroute
Double click to enter description.

Final Writer
Double click to enter description.

Check For Low Resolution Images
Double click to enter description.

Low Resolution ReRoute
Double click to enter description.

Publish/Localize Content
Double click to enter description.

Validate Locales
Double click to enter description.

Create Image Localisation Request
This step is to launch an Image localisation...

Create Content XML
Double click to enter description.

Send Content for Translation
Double click to enter description.

Create Live Copies
Double click to enter description.

Publish Localize Reroute
Double click to enter description.

Publish Article
Double click to enter description.

Drop steps or participants here

Flow End

Figure 5.18 CMS Workflow for Implementing Knowledge Article Publishing Process

- Main steps in workflow design are process modeling, user/role assignment, notification, and complex processing.
- Prerequisites for workflow design are content strategy definition, user interviews, definition and mapping of content roles, identification of workflow task items, exception handling, compiling all the best practices and checklists, and compiling all the tools and accelerators.
- High-level steps in designing workflow are to model the process steps, security assignments for workflow steps, integration, optimization, and testing.
- The chapter discussed a case study for the knowledge management workflow.

Chapter 6

Content Information Architecture, Taxonomy, and Metadata

Efficient content organization and relevant information discovery forms one of the most critical success factors for most content programs. Content organization and information discovery are also crucial elements of content strategy exercise. Information architecture, taxonomy, and metadata are disciplines that aid us in this aspect.

Content information architecture is planning, organizing and tagging content appropriately so that it is accurately searchable and easily accessible. This area primarily refers to methodologies on how information is stored and organized. Information architecture makes Web content more usable. It organizes the information intuitively to make it easy for end-user to locate and use. *Content metadata makes content more structured and discoverable. Semantic metadata tags may further enhance the meaning of content for a given context. Taxonomy stores the key terms, their definition, and relationships.* In some scenarios, metadata hierarchy would be defined within enterprise taxonomy. We have seen that information architecture, metadata, and taxonomy are involved in content strategy exercise. Hence, a content strategist may work closely with an information architect and a taxonomist, as they have great influence on effective content strategy.

In this chapter we look at various concepts of information architecture, metadata, and taxonomy and examine how they aid the content strategy. We begin by looking at goals and elements of information architecture (IA) and understand its best practices and role of IA in content strategy. We then look at taxonomy and metadata concepts, including their advantages, business drivers, and best practices. In subsequent sections we take a look at metadata standards, metadata utilities, and taxonomy governance. The chapter concludes with a case study for understanding the utility of metadata in search process and various other utilities of metadata.

Enterprise architects, content architects, content strategists, and information architects may find this chapter useful.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

6.1 INTUITIVE INFORMATION ARCHITECTURE

Information architecture (IA) generally provides the guidelines and structure to organize the Web site and its content. IA provides a structured model of the system and organizes tasks and information intuitively. It specifies various things such as content and page relationships, content hierarchy, navigation model, site taxonomy, and so forth.

Goals of IA

The following list provides the main design goals of IA:

- To enhance ease of information discovery, storage, and delivery
- To design and present a consistent and predictable navigation structure for the end-user
- To organize the information intuitively so that it is end-user-friendly.
- To enhance content and site usability and increase user productivity.
- To eliminate information redundancy and to enhance end-user experience
- To influence optimal placement of feature and functionality.

Elements of IA

IA is mainly concerned with organizing the information and uses various means to achieve it. The key IA elements (within the context of content) are:

- Storage structure for content
- Taxonomy that provides term hierarchy and term relationship for content metadata
- Metadata that ties content to its context
- Navigation models such as menus, header and footer, left navigation, breadcrumb, quick links, etc.
- Sitemap that visually organizes the site structure, page hierarchy, and relationship
- Content search that retrieves relevant information
- Content model that depicts content structure

In subsequent sections we will look at taxonomy and metadata in greater detail. Content search and content model are discussed in separate chapters.

Defining IA

The broad steps for defining IA are listed below. The steps, though aimed at defining IA for a Web site, can be used at a granular level for site content as well.

- **Identify the key stakeholders of the site and target audience:** There may be various stakeholders interested in Web site design and its content. Various business stakeholders, marketing and sales team, operations team, site administrators, and others may have a stake in the Web site. They may be interested in various aspects of the Web site, such as message communication, information access, marketing and sales, completing a task, self-service, and others. As a starting point, we need to list all those stakeholders and the intended target audience.
- **Map the stakeholders' and audience's goals to IA requirements:** For each of the stakeholders, we need to understand their goals and expectations from site content. To understand target audience needs we can conduct user surveys. We then need to map the stakeholders' goals and needs to IA requirements. A sample mapping of stakeholders' goals to IA requirements is given Table 6.1.
- **Information design:** We need to analyze the IA goals obtained in the previous step and design the IA to align with specified goals. For instance, from the content management standpoint, the following things need to be considered:
  - How can IA help users discover relevant content faster?
  - How can IA enhance content reusability?
  - How can IA help reduce content management costs?
  - How can IA optimize content workflows?

Table 6.1 Stakeholder's goals to IA requirements

Stakeholder	Goals and Expectations	IA Requirements
Business stakeholders	• Make the site easier to use • Users should discover relevant content easily and quickly • Increase site traffic • De-clutter the home page and landing page	• Simplified site structure • Faster and efficient search • Categorized content accessible through menus • Easier navigation • Easy-to-use content and Omni-channel enabled responsive content.
Operations team	• Reduce support incidents and support calls	• Self-service-enabled features
Marketing team	• Provide targeted content	• Metadata-driven information delivery • Personalized information delivery • Provide "Popular downloads," "Most discussed," "Quick links," "Saved search," and other similarly categorized content
End-user	• Easy access to information • Faster completion of tasks	• Provide quick links and FAQs for frequently accessed content • Optimize steps to discover and complete a process

These design considerations help in development of various IA elements and processes.

- **Define the IA elements:** Based on the insights gathered from IA design, an information architect can define various IA elements.
  - **Wireframes**: Wireframes provide the blueprint and structure of site pages. Wireframes depict key UI elements such as screen layout, navigation, UI components, and user controls. Although creative agency and UX designers mainly develop wireframes, information architect can also add value to this exercise. The information architect will identify key site sections, focus areas, and navigation models and map the vital functionality to such areas. For instance, the information architect can map the global search feature to a prominent site area (such as top right corner of global header) visible to all users.
  - **Sitemap**: Identify logical grouping and categories of the site sections and site content and develop a sitemap based on that. The groupings in the sitemap are normally based on user segments (such as "For customers," "For Business"), geographies (such as "US," "EMEA," "Asia Pacific," etc.), departments, content categories, and so forth.
  - **Navigation model**: Wireframes and sitemap can help the information architect to design various navigation aids such as left navigation, context menus, and breadcrumbs. In addition to these traditional navigation structures, the information architect can also develop more effective elements to help end-users access the information quickly. These structures include content modules such as "Previously viewed articles," "Related links," "Quick links," "Bookmarked links," "Most popular products," "Most liked articles," "Saved Search,", "My Favorites," "Popular discussions," and the like. Other intuitive ways to convey the relevant information quickly is through "Article summary" and "Highlights," which provide the gist of the large content. Another effective navigation element is to use horizontal and vertical navigation bars, which provide logically grouped content.

Best Practices While Defining IA

While defining the IA elements, it is suggested to follow these rules of thumb:

- The navigation structure should be consistent for all IA elements. For instance, the URL, left-hand navigation, breadcrumb, and all similar navigation elements should depict consistent content hierarchy that ensures a predictable navigation pattern for the end-user.
- IA should be designed keeping the end-user in mind. IA should help the end-user find the required information quickly and provide friendly navigation. User persona analysis should be used to understand user's navigation patterns, persona-based information consumption and preferred presentation types and to design the IA elements to satisfy user needs. Content should be targeted for specific user roles, and site channels should be based on this analysis.

- All IA elements should focus on simplifying information access, and hence they should aim to de-clutter the pages. In case of left navigation, a maximum of two levels should be used. In case of navigation bars (such as horizontal bars), the high-level options should be between three and five in number and should not have more than three levels of navigation.
- The labels for content sections and navigation elements should be intuitive, clear, and unambiguous.
- Extra care should be taken while designing home pages, landing pages, and other frequently accessed/business-critical pages. The content and navigation elements of these pages need to be simplified so that they are easily usable.
- Wherever there is excessive content or a large set of pages, try to minimize information overload on the user. This is achievable through the following means:
  - Provide bulleted summary, overview, or abstract of excessive content
  - Provide gateway pages that group the pages into logical categories so that users can quickly select the most relevant pages
  - Provide "Quick links," "Related links," "Recent searches," "Saved searches," and similar navigation tools based on user's navigation behavior or past browsing history
- While storing content in folders, top-level grouping can be based on geo or language of content. Subsequent levels should reflect the site sections and/or usage scenarios. We can store common and reusable content in the "commons" folder.
- Design and test the IA on all target platforms and devices and ensure that it provides a consistent user experience on all channels and devices.

Role of IA in Content Strategy

The information architecture in a content scenario has two parts: internal information architecture and external information architecture.

- Internal Information Architecture represents the content type definitions and is expressed through the following artifacts:
  - Content model (or content blueprint): We discussed the content model in detail in Chapter 2 as one of the core elements of content strategy.
  - Functional taxonomy (or metadata definition): The taxonomy provides the content tag and metadata term hierarchy.

  These elements are critical to establishing content definitions and relationships that can be presented in many different ways, including but not limited to the more current Web site information architecture.
- External information architecture focuses on the site organization in terms of pages and page hierarchy as well as the layout and structure of any given page. Common presentation templates, reusable components, navigation, and overall sitemap are part of this.

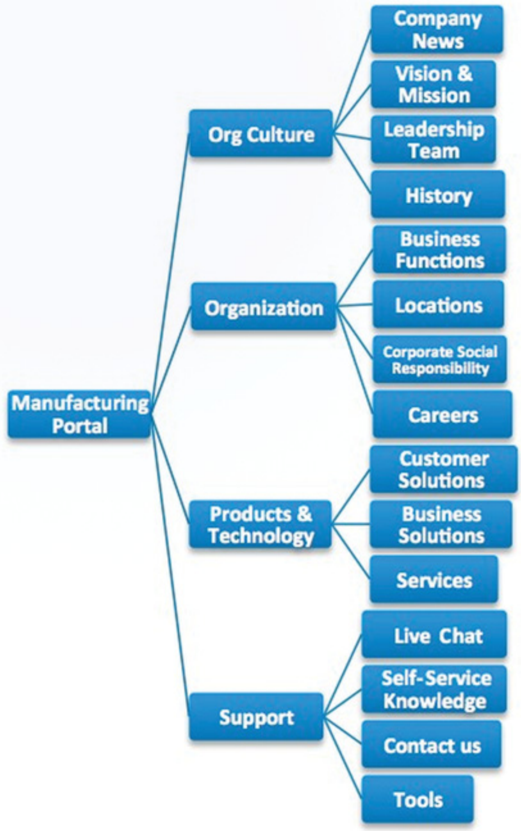


Figure 6.1 Content IA for Manufacturing Portal

IA fulfills some of the key goals of content strategy, such as enhanced content usability and enhanced content discovery.

IA Design Samples

We have looked at the core concepts of IA from the content standpoint. In this section we look at two sample IA designs and understand the contribution of IA to content organization.

Content IA for external facing manufacturing portal In case of manufacturing portals, the main goals for online channel is for optimal information delivery and actively engage end-users through various tools.

In Figure 6.1, we have given a simple content hierarchy for manufacturing a domain portal.

The site content is mainly organized into four sections: org culture, organization, products and technology, and support. Within each major content section there are various subsections that logically belong to that category. A brief look at the purpose served by each of these content sections is given below:

- Org culture: This section consists of content that conveys information about the organization, its culture and policies, along with vision and mission statements. The main purpose is to spread awareness about the organization and its history.
- Organization: In this section users can find the high-level hierarchy of the organization through its business functions and units, as well as information about its location. Organization can also convey its social commitment through Corporate Social Responsibility (CSR) programs and announce current openings in this section.
- Products and technology: This is one of the core content sections for a manufacturing organization. This section contains products and solutions categorized by various logical categories so that users can easily locate the most appropriate product. It usually contains a descriptive search feature to search for products and services.
- Support: This section contains various tools and content to enable the self-service model. Users can use chat and solution knowledge base and tools to complete their tasks or to seek support. The goal of content is to help users resolve their issues faster and to facilitate user engagement with support personnel with fewer clicks.

Content IA for internal employee portal Figure 6.2 shows the content IA for an internal portal. For this example we have taken an employee portal scenario. The key design goals for the employee portal are self-service and improvement in productivity. Hence, the IA is designed to fulfill these goals.

The content is organized to enable employees to perform their tasks most efficiently. Key content categories are explained below:

- Employee needs: Employees can access various applications such as payroll and benefits in this section, and it should provide easy access to policies and a handbook.
- Learning: Content and tools in this section should help employees self-train by providing technical documents and training content.
- Tools: Should provide all necessary tools and processes that may improve employee productivity.
- Contacts: This section provides the key contact for various needs.
- Useful links: This section should provide most popular links and links to various utilities. Additionally, it should also allow employees to bookmark their favorite links.
- Collaboration: This section should provide employee engagement and encourage active participation from employees. This section would provide blog, wiki, and forum to enable collaboration among employees.

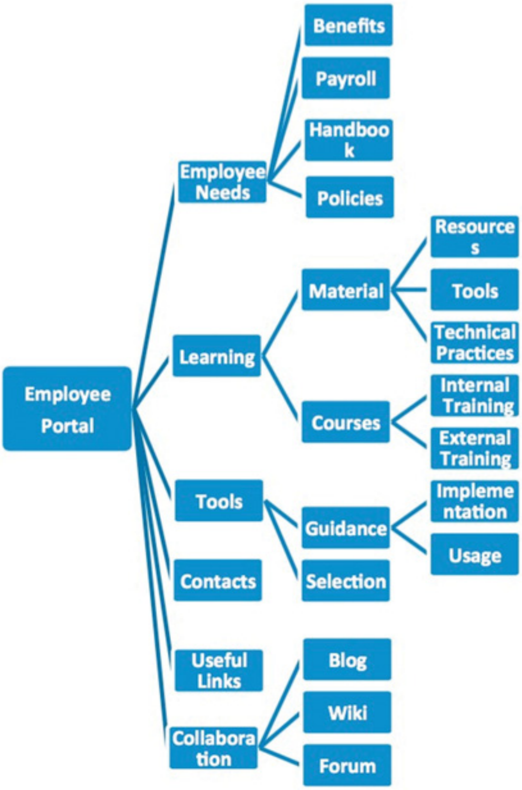


Figure 6.2 Content IA for Internal Employee Portal

6.2 INTRODUCTION TO TAXONOMY AND METADATA

In this section we discuss key concepts of taxonomy and metadata from the content standpoint. Information architecture mainly organizes the information and helps in information discovery at the front end (covering presentation elements, sitemaps, etc.), and taxonomy/metadata serves the same purpose at the back end (through tagging, categorizing, etc.)

Taxonomy is a hierarchy of controlled vocabulary values that can be used to organize, tag, and classify content. Taxonomies are essential for both content management and enterprise search. Taxonomy helps us catalog content and data into logical categories so that they are easily accessible. At a general level, taxonomy in the context of content management refers to the content hierarchical structure (along with its URL structure and site organization) and the controlled vocabulary values (including look-up lists, non-modifiable lists, etc.). Taxonomy describes each term through title, label, synonym, metadata, relationships, and other identifiers. Hence, taxonomy is widely used as a system or record (SOR) for metadata terms.

Metadata, which provides data about data, is an additional property of content, which basically describes content. Metadata is normally tagged with content.

Metadata describes content, its behavior, and its key attributes. It helps in organizing, accessing, and categorizing content and makes content consistent across various sources, which makes it easily searchable and findable. For example, product type and product location form metadata for product content.

Metadata provides consistent, understandable, and searchable information about content and its structure. Metadata is normally added during the content creation phase. The primary use of metadata is in finding relevant content for a given context. Content metadata can also be used to enhance content reusability. Following are some examples of content metadata: region, language, title, author, keywords (controlled vocabulary), security tag, workflow status (draft, ready for review, in review, approved, ready for approval, ready to be published, published, archived, ready for translation), and tracking/auditing metadata (author, date, modified, reviewer, approver, time, geo, translation status, archival status, location).

Relationship between taxonomy and metadata Essentially taxonomy provides the value for metadata. For instance, if "language" is a metadata attribute for content, then "English" is one of the values for "language" attribute. "English" is a controlled list of values, which comes from enterprise taxonomy that maintains a list of all supported languages in a specific hierarchy.

Advantages of Taxonomy and Metadata

The main advantages of taxonomy and metadata for digital content management are as follows:

- Enhances the content structure and organizes it, making it more accessible and manageable.
- Provides personalized and contextualized content for the target audience, leveraging security and preference-related metadata and taxonomy values.
- Makes content and asset resources usable and readable by other systems.
- Enhances sharing, exchanging, and interoperability of content and assets.
- Helps in establishing the relationship among content data and helps in its classification
- Helps in easy and fast discovery of relevant content and improves content accessibility. Metadata helps in content findability and plays a key role in information discovery. It marries content to meaning and increases search effectiveness. Presentation platforms and search engines can use metadata to retrieve relevant content.
- Helps in logical categorization of content. Metadata can be used to attach categories to the topics.
- Metadata is key for an enterprise search function. An enterprise search engine uses metadata to identify the relevancy of content. Metadata adds meaning to the information, thereby enabling efficient search. Taxonomy and metadata

attributes also help in faceted search (also referred to as guided navigation). We can filter content based on its metadata.

- Plays a key role in document management by helping identify, index, and search the documents.
- Helps reuse content across various page sections, devices, and geographies.
- Reduces duplicate content and hence reduces the overall cost and effort of content management.
- Plays a key role in improving workflows. For instance, the "content workflow status" metadata value can be used by workflow engine for progressing the workflow automatically and for reporting/auditing purposes. For example, when the content workflow status changes from "Ready for review" to "Reviewed," it progresses the workflow to its next step.

Business Drivers for Taxonomy and Metadata

The key drivers for enterprise taxonomy and metadata are:

- Standard content classification: Due to the presence of multiple content systems, it is important to provide a uniform classification mechanism for enterprise content. This allows for easier and faster content discovery and avoids duplicate content. Automatic classification/reclassification of content during ingestion using enterprise taxonomy enhances content accessibility and discovery.
- Productivity enhancement: Enhances user productivity due to faster information discovery enabled by content classification.
- Efficient content findability: The enterprise taxonomy terms may help in content tagging, content search, and similar processes. Some of these terms can be used as markers for automatic routing (in content workflow) and further content classification.
- Uniform tagging: Centralized metadata management system provides uniform and consistent tagging across enterprise-wide content.

Taxonomy and Metadata Best Practices

The key taxonomy and metadata best practices are as follows:

- An enterprise-wide centralized taxonomy should be defined and managed by a taxonomy management tool.
- It is preferred to have a centralized taxonomy and metadata management system to avoid term duplication.
- Metadata should be based on standards such as Dublin core standard (discussed in Chapter 7).
- Automatic content/data categorization and metadata assignment should be provided wherever possible.

- Metadata governance process is needed to specify the usage and management of metadata.
- CMS and search systems should be integrated with taxonomy and metadata systems for tagging and searching content.

Types of Metadata

In the context of content, we can identify two broad types of metadata: system-level metadata and application-level metadata

System-level metadata values are used by a content system for content management and identification. Normally these are similar to audit values and are non-modifiable. Metadata such as created-by, updated-by, deleted-by, creation-time, deletion-date, and workflow-state are internally used by CMS to manage content. Usually these values are hidden from authors during content authoring and feature as read-only values in audit reports.

Application-level metadata are the ones that a taxonomist can define and model based on the application needs. A taxonomist (in collaboration with a content strategist and business stakeholders) can build a hierarchy of metadata needed for application content. Content-authoring templates allow the content author to specify the application metadata during authoring. This includes metadata such as language, region, product name, product family, and others that can accurately describe content.

With the emergence and popularity of Web 2.0 technologies, content sharing and content syndication have gained popularity; in such cases end-users can tag content through social tagging. *Social tagging is used for social content and user-generated content (UGC).* Usually end-users can select the tags or create new tags for their content. We will take a closer look at these in the upcoming sections.

Some metadata systems also classify metadata into other categories such as administrative metadata (such as backup date, cache creation date, server start time, etc.), business metadata (such as target business audience, campaign category, etc.), workflow metadata (such as workflow state, workflow owner, etc.), search metadata (such as search keywords, search index name, etc.), technical metadata (automatically determined dynamic data such as size, resolution, dimension, bit ratio, etc.), security metadata (such as access role names, access permission values, etc.), and descriptive metadata (which describes the digital asset).

In the next section we will look at the process to model the application metadata needed for the solution.

Metadata Hierarchy Modeling

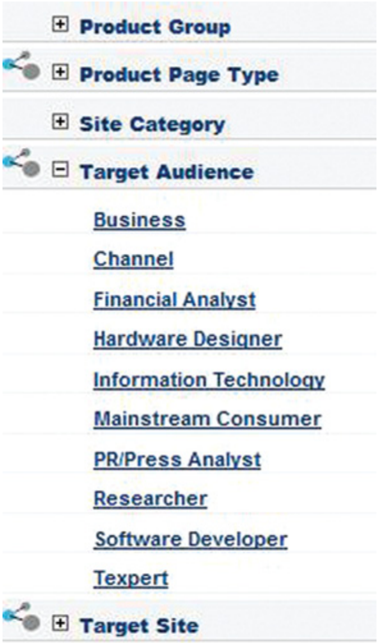
The first step in creating a metadata hierarchy is to understand the terms and values needed for the application. A taxonomist along with a content strategist will lead this exercise. An enterprise-wide taxonomy may contain the terms needed for various enterprise applications. We can update them as needed by the application and reuse wherever required.

System-specific metadata is owned and managed by CMS. For modeling application-specific metadata we need to understand the business terms, content hierarchy, and interrelationships and model them in taxonomy. The basic properties or attributes for content become its metadata. For example, language and target audience become metadata attributes for Web content targeted across multiple geographies; product attributes (such as model, title, and name) become metadata for product content.

In many cases an existing content hierarchy can be used for metadata hierarchy modeling as well. For instance, in an e-commerce solution domain, the product hierarchy is fairly well structured. We can model the product metadata using the same hierarchy.

Taxonomy contains facets (or groups) for modeling term hierarchies and for term categorization. Common taxonomy facets are business domain (for modeling common business domains such as finance, marketing and sales), industry (for modeling industrial domains such as retail, manufacturing, life sciences, education, and the like), geography (for categorizing terms belonging to a region, country, or city), language (to group languages), roles (to model various user roles), content format (to depict various formats such as text, video, image, etc.), and audience (to depict target audience). Taxonomy facets also vary based on the enterprise domain. For an e-commerce organization, products and its related attributes (such as product model, product price range, etc.) may form important facets. Facets and taxonomy terms can be used as content metadata values during content creation.

An example of target audience metadata is shown in Figure 6.3.



⊞ Product Group
⊞ Product Page Type
⊞ Site Category
⊟ Target Audience
Business
Channel
Financial Analyst
Hardware Designer
Information Technology
Mainstream Consumer
PR/Press Analyst
Researcher
Software Developer
Texpert
⊞ Target Site

Figure 6.3 Target Audience Metadata

The values for "Target Audience" metadata come from taxonomy terms, as they are controlled values applicable across enterprise.

Functional taxonomy

We can create a functional taxonomy by closely modeling based on the organization functions and business hierarchy. In an e-commerce domain, functional taxonomies are hierarchies of product categories and subcategories through which content is organized in a repository. The categories in this taxonomy provide a logical grouping of the content. A sample functional taxonomy is depicted in Figure 6.4.

In Figure 6.4, "Pages" are organized into various categories such as "Detail," "Downloads," "Events," etc. These categories depict the site sections on which its constituent content will be used. Content chunks, assets, and other elements can be categorized in a similar fashion.

Once the metadata hierarchy is modeled in a taxonomy, it needs to be reviewed and approved by all concerned stakeholders. A taxonomist should establish a governance process to update the taxonomy on the need-to basis.

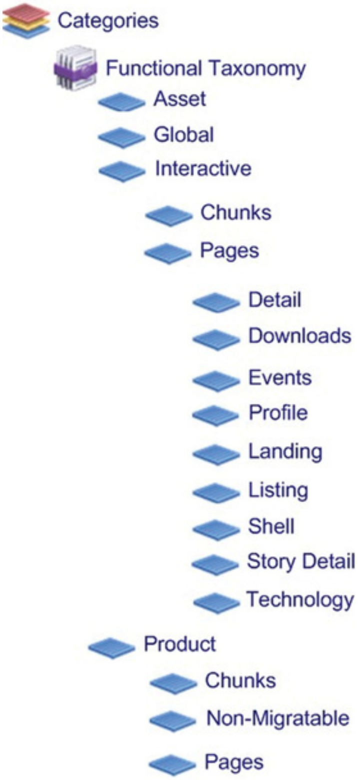


Figure 6.4 Sample Functional Taxonomy

6.3 METADATA USAGE IN RELEVANT CONTENT DISCOVERY

This section presents the general steps depicting the role a taxonomy plays in content rendition and relevant information discovery. The process described in Figure 6.5 is for content search.

As a prerequisite, the taxonomist has created a hierarchy of terms and controlled vocabulary through a tagging interface. These terms are incrementally updated based on inputs from other systems and from business.

In this example we are assuming that CMS supports built-in tagging and metadata interface and we will only add the taxonomy terms as values for tagging.

Content systems are integrated with a taxonomy, and the taxonomy terms are used for tagging content. Content-authoring templates offer provision to add semantically relevant metadata terms while creating content. Many CMS have built-in mechanisms to store the associated metadata along with its content. Metadata will also be published along with the content. Published content (in XML or HTML format) also contains the metadata.

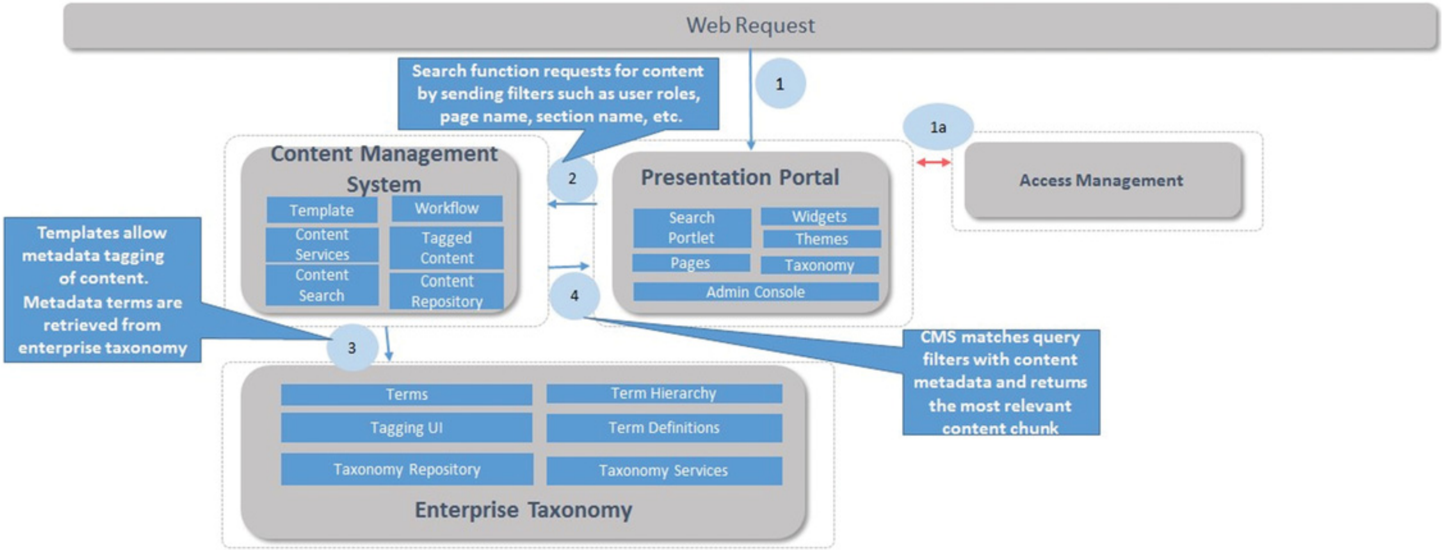
When the user searches for content in a portal, the application logic sends the context as additional filters (such as user role, language, site section, page name, etc.). These filters are matched with metadata values to find the most relevant content for the given context. Matched content will then be rendered in a presentation portal.

6.4 INTEGRATION OF METADATA WITH CMS

In this section we take a look at integration aspects of metadata with CMS.

Built-in CMS metadata support Almost all CMS systems provide basic support for metadata. They provide features to tag the metadata values with content and support metadata creation. The key touch points of content and metadata are:

- **Content authoring and editing**: Content-authoring templates provide fields to specify the metadata value during content authoring and editing. In addition to the author-specified metadata, CMS may also store system-level metadata attributes such as content created date, version number, content version date, and others.
- **Content workflow**: Each step of the workflow automatically updates the metadata value (for instance, when content is approved, its "workflow status" changes from "Under review" to "Approved"). In many cases the metadata is used by the workflow engine to route content to the next workflow step. (For instance, the metadata change from "workflow status" = "authoring" to "workflow status" = "Ready for review" will trigger the workflow engine to advance the workflow to the next step, routing it to the inbox of the configured reviewers.)



Search function requests for content by sending filters such as user roles, page name, section name, etc.

Templates allow metadata tagging of content. Metadata terms are retrieved from enterprise taxonomy

CMS matches query filters with content metadata and returns the most relevant content chunk

Figure 6.5 Role of Metadata in Content Search

- **Publishing**: During publishing, the publishing engine may also publish the configured metadata to the target environment. If the publishing environment is a mirror replica of the authoring environment, then the metadata may be copied over to the target environment. If the target environment is a Web server or a file server, the metadata/tag information will be embedded along with content/HTML.
- **Localization**: During localization we may need to configure the locale-specific metadata related to language, geography, region, and country. The metadata module within CMS should be able to support this.

Metadata is also used in content translation, content archival and during content transformation.

External metadata system integration In large enterprises, enterprise-wide metadata will be centrally managed within a metadata management system (MMS). MMS holds and manages metadata needed for enterprise-wide applications and has strict governance processes to update the values. In such cases CMS needs to be integrated with MMS. CMS integration with MMS is discussed in Chapter 7.

6.5 METADATA STANDARDS AND FORMATS

In this section we discuss some of the key industry standards with respect to metadata. Adopting these standards helps greater interoperability between internal and external applications. Dublin Core and SKOS are two standards that can be used for this purpose.

Dublin Core

Dublin core is one of the most popular metadata standards that provides interoperable metadata standards and helps in describing digital content and making it discoverable. Dublin core metadata terms can be used as vocabulary terms to describe the content chunk and assets.

Dublin core provides 15 optional metadata elements that can be used to describe resources through metadata elements, such as *dc.title, dc.subject, dc.creator, dc.type, dc.format, dc.source*, etc. Dublin core provides simple yet effective way of adding content metadata to prepare the semantic Web page. It also helps in achieving SEO (mainly for internal search engines).

Content and Dublin core

We can use Dublin core metadata for tagging content chunks to add a semantic description of the content along with comments and other attributes.

A sample Dublin core metadata for product overview content chunk (for HTML or XHTML format) is given below:

```
<meta name = “dc.language” CONTENT = “en_us”>
<meta name = “dc.title” CONTENT = “Product A Overview”>
<meta name = “dc.type” CONTENT = “Text”>
<meta name = “dc.title” CONTENT = “Product Title”>
<meta name = “dc.keywords” CONTENT = “product A, overview”>
<meta name = “dc.subject” CONTENT = “Test subject”>
<meta name = “dc.description” CONTENT = “The content
provides a detailed overview of product A”>
```

For XML-based content, we can use the Dublin core metadata in a format depicted in Figure 6.6.

CMS support for Dublin core

Popular CMS products provide support for Dublin core metadata, and it is mainly used for search-based information discovery and for internal SEO purposes. Drupal CMS helps the users configure “Dublin core settings” through metatags module. Besides this, there are other Drupal modules (such as Drupal core metadata module) that help the administrators add metadata to Drupal content.

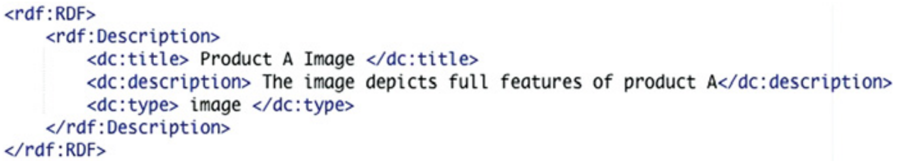
There are WordPress plugins such as Dublin core metadata Plugin that can be used to add Dublin metadata to WordPress posts.

Simple Knowledge Organization System

Simple Knowledge Organization System, or SKOS, was designed mainly for structuring and exchanging knowledge assets. SKOS is a W3C standard that uses RDF, RDFS, and OWL specifications to model controlled vocabularies. SKOS is one of the key elements of semantic Web. SKOS provides a structured and powerful framework for expressing knowledge organization systems in a machine-understandable way. SKOS models various structures such as vocabularies, taxonomies, dictionaries, classifications, glossaries, and others. It describes a term along with its attributes and defines its parent-and-child relationship.

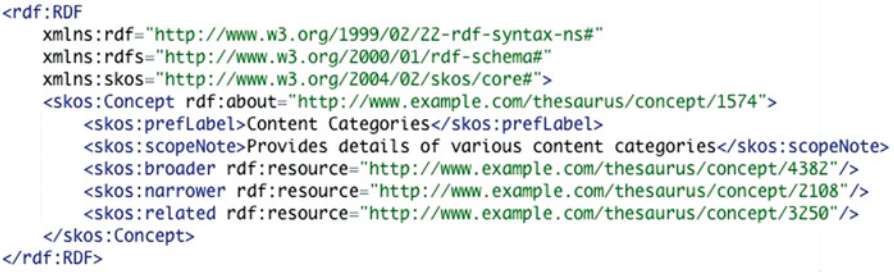
SKOS vocabulary mainly involves terms to describe a concept, such as *skos:concept, skos:prefLable, skos:broader, skos:narrower, skos:related*, etc.

A sample SKOS for defining the content categories hierarchy is given in Figure 6.7.



```
<rdf:RDF>
    <rdf:Description>
        <dc:title> Product A Image </dc:title>
        <dc:description> The image depicts full features of product A</dc:description>
        <dc:type> image </dc:type>
    </rdf:Description>
</rdf:RDF>
```

Figure 6.6 XML-Based Dublin Core Metadata



```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:skos="http://www.w3.org/2004/02/skos/core#">
    <skos:Concept rdf:about="http://www.example.com/thesaurus/concept/1574">
        <skos:prefLabel>Content Categories</skos:prefLabel>
        <skos:scopeNote>Provides details of various content categories</skos:scopeNote>
        <skos:broader rdf:resource="http://www.example.com/thesaurus/concept/4382"/>
        <skos:narrower rdf:resource="http://www.example.com/thesaurus/concept/2108"/>
        <skos:related rdf:resource="http://www.example.com/thesaurus/concept/3250"/>
    </skos:Concept>
</rdf:RDF>
```

Figure 6.7 SKOS with Content Categories

We will discuss SKOS in more detail while exploring “Content formats” in Chapter 7.

Content systems and SKOS

In the content management scenario, SKOS is mainly used to model the taxonomies and controlled vocabulary. A presentation engine can model the entire content page hierarchy in SKOS taxonomy. It can then depict the related content chunks on the page and child pages (using narrower term relationship) and parent page (using broader term relationship) attributes of SKOS.

CMS support for SKOS

Content management systems mainly use SKOS for defining controlled vocabularies, tags, and content categories. Drupal has an SKOS importer module to import SKOS thesauri into Drupal as taxonomies/terms. PoolParty Thesaurus Wordpress plugin makes the Web site more understandable by using SKOS-based controlled vocabulary. Vocabullary terms are used for term-based article discovery. Climate change glossary plugin also allows import of SKOS thesaurus into Wordpress blog.

Let us now look at a case study to depict the usage and effectiveness of metadata in a content scenario.

6.6 CASE STUDY: CONTENT METADATA TO INCREASE SEARCH EFFECTIVENESS

A uniform and consistent metadata strategy is one of the vital elements for increasing search effectiveness. Let us look at this in detail in this section. We will look at both internal search and external search scenario for an e-commerce content.

Internal Search

System-level and application-level metadata helps in internal search. Let us take a simple example of the search for a product description content chunk using product name as the query term. The content author may select the appropriate authoring template to create content. CMS will automatically populate the system-level metadata values such as created by, creation date, and others.

Once product content is authored, the author will tag the application-specific metadata. In this scenario the applicable metadata would be:

- Product line = Television
- Product family = LED Television
- Product model = Vseries
- Product name = 56LEDV

Once product content is reviewed and published, it is stored in a content repository along with associated metadata. When the end-user searches for product information using various filters, the search interface uses the tagged metadata to retrieve the relevant information. For instance, if the search query is "List all LED Televisions released since last 3 months," search query filters will internally use the metadata as follows:

Fetch all content with metadata: *Product line = Television and Product family = LED Television and (current_date - created_date) < 3 months*.

Note that the filter will also use "*created_date*," which is a system-level metadata.

In addition to the regular metadata values, the metadata system/component can also capture additional information about the content. Let us look at the following metadata:

- Product line = Television
  - Synonyms: TV, Television set, display device, display screen
  - Related Terms: Smart TV, Gaming Console, Blu-ray Player, Home Theater system
  - Broader Terms: Electronics, Consumer products,
  - Narrower Terms: LED TV, LCD TV, Plasma TV

The metadata information about "Synonyms," "Related Terms," "Broader Terms," and "Narrower Terms" can be used by search engines to improve search efficiency. For instance, the search engine can bundle the sales offers using products in "Related terms" and "Narrower terms" category. The search engine can show "Related searches" based on "Related terms." It can also expand the search of user's query term along with all configured synonyms in a taxonomy for improved search results.

External Search

For external search engines like Google, Bing, and Yahoo, metadata is one of the important SEO parameters. Some of these aspects figure in the SEO discussion.

From the content standpoint, page-level metadata plays a vital role in helping external search engines:

- Page-level keywords
- Metatags
- Title

A more detailed discussion of SEO can be found in Chapter 10.

6.7 OTHER UTILITIES OF CONTENT METADATA

Let us look at other advantages of metadata in content scenarios.

Metadata-Based Content Categorization

Though CMS provides sophisticated categorization features, metadata can also help in finding the logical category for a given content piece. A highly consistent and controlled metadata would also provide an accurate content categorization. For instance, if the product content has metadata values Product line = Television, that product content would logically fit into the categories such as “Television,” “Consumer Electronics,” and the like.

The strength of metadata-based content categorization may be more visible in case of user-generated content (UGC). In case of UGC, content will be tagged by end-users, and they may choose either to use an existing tag name or to create a new one. By analyzing the content tag values, CMS can categorize the content into its logical categories.

Marketing and Sales Support

Metadata can be used for delivering targeted, personalized content and help in upselling and cross-selling. Metadata attributes such as target audience, geography, language, and country can be used to target the most relevant content to the target audience. As we have seen in the case study, related terms, broader terms, and narrower terms can be used to bundle the products to cross-sell and upsell.

Metadata-Driven Content Personalization

Personalization is one of the primary utilities of metadata. User attributes such as user role, location, preferences, interest categories, date, time, purchase history, and others can be tagged as application-level metadata, and this information can be used to filter content and deliver the most relevant parts of it to the end-user.

Metadata-Based Page Customization

A content administrator can also customize the system by configuring things such as page layout, landing page, number of page sections, and similar information. This can be stored as metadata at the page level and can be used for page rendition.

The end-user can specify the preferences such as the number of content sections, content category that can be stored as end-user-level metadata information, and can later be used for customizing the user experience.

Content Metadata for Navigation

Sometimes metadata information can also be used to dynamically generate the navigation. If there is a well-defined metadata hierarchy, then we can automatically construct the navigation using the metadata. Let us consider this metadata hierarchy (starting with the broadest and ending with the narrowest):

Product Line (e.g., Television) → Product Family (e.g., LED TV) → Product Model (e.g., M5291)

With this hierarchy, when the user visits the Television page/content, the system can show the child links to LED TV because it can derive this relationship from the metadata hierarchy.

Analytics Metadata

Metadata values such as target audience, user segment, download count, and page views can be populated and used by a Web analytics engine to provide insights into user behavior and content usage and effectiveness.

Content Metadata for Workflow

One type of system-level metadata is related to workflow stages. As content navigates from one workflow step to another, the corresponding metadata (such as content status, workflow status) is updated. The workflow engine can use this metadata to route the content to an appropriate inbox.

Using Metadata for Reusing Content Chunks

Let us consider a simple example to demonstrate the chunk reusability that can be achieved through metadata. Metadata can be used to achieve both authoring time (design time) chunk reusability and runtime chunk reusability.

Let us say that “Product family overview” content is already authored for product1. The metadata for that content chunk is as follows:

ProductFamily = ”Converter” ContentType = family_overview, productname = product1

Now, if want to author content for product2, which belongs to same family, we can fully reuse the family_overview content of product1, as both products belong to same family. So in the authoring template based on the metadata *ProductFamily = ”Converter” and ContentType = family_overview,* the authoring template can be configured/customized to automatically select the matching content chunks tagged with this metadata. The author can select the most appropriate one and reuse it for the current context.

At runtime, the presentation engine can use the page-level metadata to dynamically assemble all relevant content chunks (with matching metadata).

Security Metadata

Security metadata values such as user roles, user groups, and permission values can be used to filter content and enforce security policies and fine-grained/role-based content access. Security metadata can also be used for content personalization.

6.8 TAXONOMY GOVERNANCE

Taxonomy governance involves processes to create, maintain, and update the taxonomy terms on a periodic basis. High-level process steps for taxonomy definition are depicted in Figure 6.8.

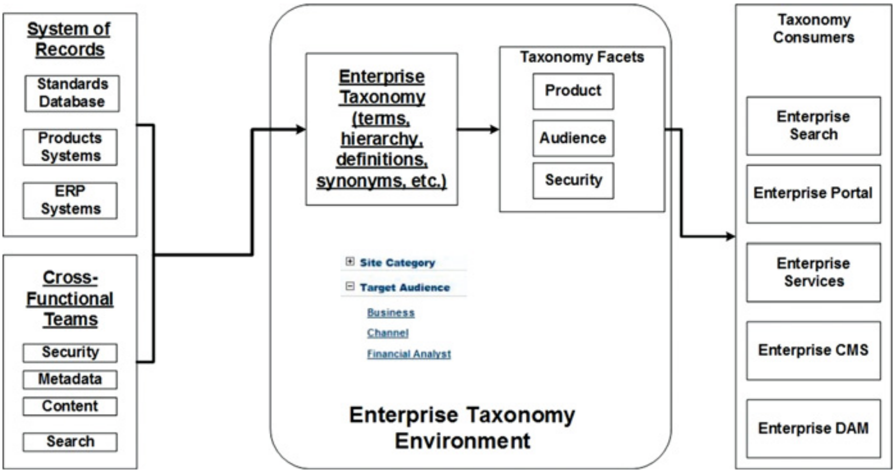


Figure 6.8 Taxonomy Definition Process

Enterprise taxonomy is usually built iteratively and fine-tuned so it can unambiguously identify all enterprise content.

We need to get the inputs from cross-functional teams such as security team, product team, subject matter experts, business stakeholders, content administrators, and other relevant stakeholders. Since taxonomy covers multiple dimensions and caters to a wide variety of enterprise applications, it is important to get inputs from all stakeholders. A taxonomy administrator can use the tagging interface to incrementally update the terms, synonyms, term hierarchy, definitions, rules, and related taxonomy data.

Enterprise taxonomy can also aggregate the controlled vocabularies from other enterprise sources such as system-of-record (SOR), standards repositories, and other internal systems. Once the term vocabulary is aggregated from various sources, a taxonomist may refine the hierarchy and add more details to taxonomy terms (such as synonym, definition, relationships, metadata, and other attributes). At this stage all the taxonomy terms are structured and semantically organized into logical hierarchies.

Taxonomy terms can be grouped into various taxonomy facets and then published to various enterprise systems such as CMS, DAM, portal, application, enterprise search, and others.

For subsequent updates a taxonomist may update or add the term into the most appropriate hierarchy.

Social Tagging

Unlike enterprise-controlled taxonomy and metadata strategy, end-users drive social tagging. This is mainly seen in social and collaborative content and user-generated content such as blogs, wiki, articles, posts, photos, and videos wherein the user tags the content with relevant keywords. The tags will evolve along with the content creation. Initially the tagging interface can provide organization's controlled vocabulary for the users and it should also allow users to add new tags. Social tagging interfaces also allow the user to re-tag existing content, create their own tags, localize tags, create tag cloud, and add multiple tags to content.

Similar to metadata, social tags mainly help in organizing and discovering relevant content. Social tags describe the core subject matter of content, content category, intended target audience, geo-location, and similar content-related information. Content platforms can not only search using social tags but also leverage tags for getting information for social analytics, popular content, most shared content, most viewed content, alerts, reporting, trending topics, most emailed content, etc.

6.9 CHAPTER SUMMARY

- Information architecture (IA) provides structure and organization for Web site content.
- The key elements of IA are taxonomy, metadata, navigation models, sitemap, content search, and content model.

- For defining IA we need to identify stakeholders, map stakeholder goals to IA requirements, design information, and then define IA elements.
- Taxonomy is a hierarchy of controlled vocabulary values, which can be used to organize, tag, and classify content.
- Metadata describes content and enhances the usability and discoverability of content.
- Taxonomy provides metadata values from its controlled vocabulary.
- Taxonomy and metadata helps in providing context-relevant personalized content and helps in faster content discovery. It also helps in logical content categorization.
- Key drivers for taxonomy and metadata are uniform classification, productivity, content tagging, and centralized metadata management.
- Main types of metadata are system-level metadata and application-level metadata.
- During content search, we can match the search query terms with content metadata to find the most relevant content.
- Most CMS systems offer built-in metadata support. Templates allow metadata tagging.
- Dublin core is one of the popular metadata standards that provide vocabulary terms for tagging content chunks and assets. It provides metadata elements such as *dc.title*, *dc.subject*, etc.
- SKOS provides an efficient way for structuring and exchanging knowledge assets. SKOS models vocabularies, taxonomies, dictionaries, classifications, glossaries, etc.
- Metadata can also be used for content categorization, marketing and sales support, content personalization, page customization, navigation, analytics, workflow, and content chunk reusability during authoring and at runtime.
- Taxonomy governance provides structured processes to continuously update the vocabulary terms in the centralized taxonomy.

Part 2

Advanced Content Management

Chapter 7

Content Integration and Content Standards

Integrations play a vital role in adding value to enterprise information. Enterprise applications aggregate data and content from multiple sources to present useful and contextual information to the end-user. Platform-agnostic content integrations, content mashups, and content aggregation portals rely heavily on integration. Through integration we can enhance the CMS functionality and augment it with additional features to fill the gaps.

In this chapter we explore the main integrations for content management systems. We take a look at the integration view and main integration requirements of a content management system (CMS) and how those integrations can be done optimally. The main intent of the chapter is to understand the key content integration techniques, best practices, and optimizations that can be achieved during integration. We take a look at integration of CMS with security systems, translation management system (TMS), search engines, portals, presentation engines, metadata management systems (MMS), and CMIS. The second part of the chapter discusses various content standards. Awareness of content standards is important to design optimal content exchange formats for a given scenario. The intent of this section is to understand content and data standards used in integrations. We discuss various formats such as XML, HTML/XHTML, DITA, SCORM and JSON along with their usage scenarios. We also look at RSS and ATOM feed formats. At the end we also discuss Web service formats such as SOAP and REST.

Content architects, integration architects, CMS administrators, CMS developers, and enterprise architects will find this chapter useful.

7.1 CONTENT INTEGRATION REQUIREMENTS

For a typical enterprise-wide content management system, the main requirements for integration fall in the following areas:

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.
© 2017 by the IEEE Computer Society, Inc. Published 2017 by John Wiley & Sons, Inc.
Companion website: www.wiley.com/go/shivakumar/enterprisecontent

- **Security integration**: The content management system needs to be integrated with the enterprise user registry (such as LDAP or active directory). Additionally other security systems such as single sign-on (SSO) systems, user access management systems (user roles and permissions), and user provisioning systems also need to be integrated wherever required. This is required to provide role-based access to content, content functions, and content resources in the CMS.
- **Content translation system integration**: Content needs to be localized for geographically distributed applications. For this, CMS has to be integrated with a translation management system (TMS). Normally translation is a part of publishing workflow or implemented as a separate workflow.
- **Search integration**: This is one of the key requirements for an enterprise. In order to enhance content discoverability and to make content available in the right place, enterprise search engine should be able to crawl and index the published content.
- **Services integration**: CMS depends on internal or external services (such as content validation services, taxonomy service, etc.). CMS also needs to expose the key content functions as services for external systems. This includes exposing content queries, content publishing, content workflows, and other key functions as content services.
- **Asset publishing system integration**: Though some CMS have built-in asset management capabilities, for robust digital asset management, CMS has to be integrated with enterprise-wide digital asset management (DAM) systems.
- **Metadata management system integration**: For tagging content and adding metadata for content, the CMS has to be integrated with a centralized metadata management system.
- **Social and collaboration integration**: Social and collaboration platforms are widely used to actively engage customers. CMS should be integrated with these systems to push authored content such as marketing content to these platforms.

7.2 CMS INTEGRATION VIEW

We will understand the big picture of the content ecosystem in this section. Figure 7.1 depicts the integration view of CMS with various ecosystem elements.

CMS platform stays at the heart of the content ecosystem. CMS provides templates, workflows, metadata, and other features to manage content. Authors and publishers use CMS for creating and managing content. CMS pulls or pushes the assets from DAM. Creative agency and designers use DAM to manage digital assets. CMS deploys content and assets to the production live site mainly through publishing workflows.

CMS gets content from various enterprise systems mainly in two ways apart from content authoring: migration and integration. During the content consolidation exercise, content from various content and document repositories is migrated to the

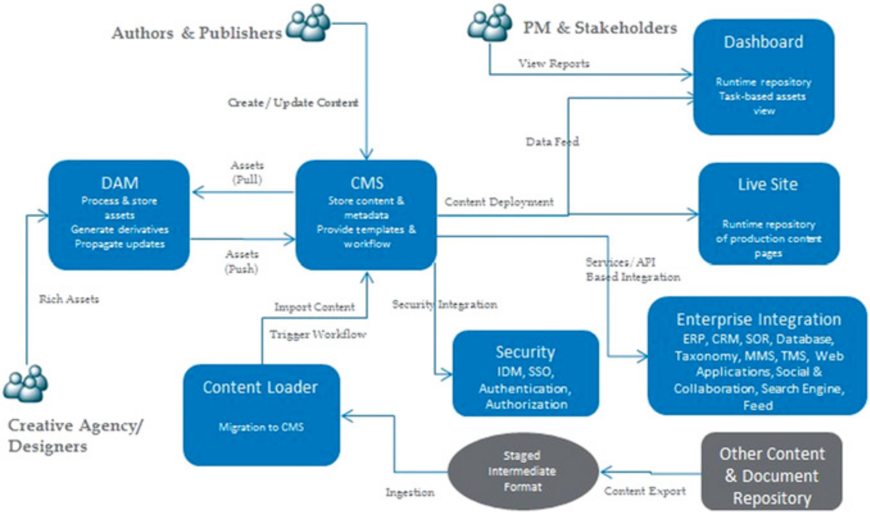


Figure 7.1 CMS Integration View

CMS through migration tools. CMS is integrated with various enterprise interfaces such as search engines, portals, MMS, TMS, Feeds, ERP, CRM, Web applications, social and collaboration platforms, and the like. The integration occurs mainly through services and APIs. We will look at various integration methodologies in the upcoming sections.

CMS will also be integrated with enterprise security systems for security use cases such as authentication, authorization, and single-sign-on (SSO). Project managers and stakeholders periodically monitor various tracking metrics, KPIs, and system health through dashboard views.

Enterprise CMS: The Big Picture

In a typical enterprise environment CMS is at the heart of Web content delivery. Figure 7.2 depicts a complex enterprise scenario that uses API gateway and ESB for integration with enterprise systems.

In Figure 7.2, the CMS authoring system is flexible in deploying the approved content to both its own publishing server (also known as a delivery system) and to other Web servers (which uses AngularJS/HTML 5 technologies). This flexible model brings robustness in the delivery mechanism, as we have an option to use core content in various contexts.

CMS is integrated to enterprise applications through API gateway layer and middleware layer. An API gateway layer provides API governance and API transformation features, and a middleware layer mainly handles orchestration and routing. This loosely coupled architecture brings robustness and extensibility to the enterprise ecosystem.

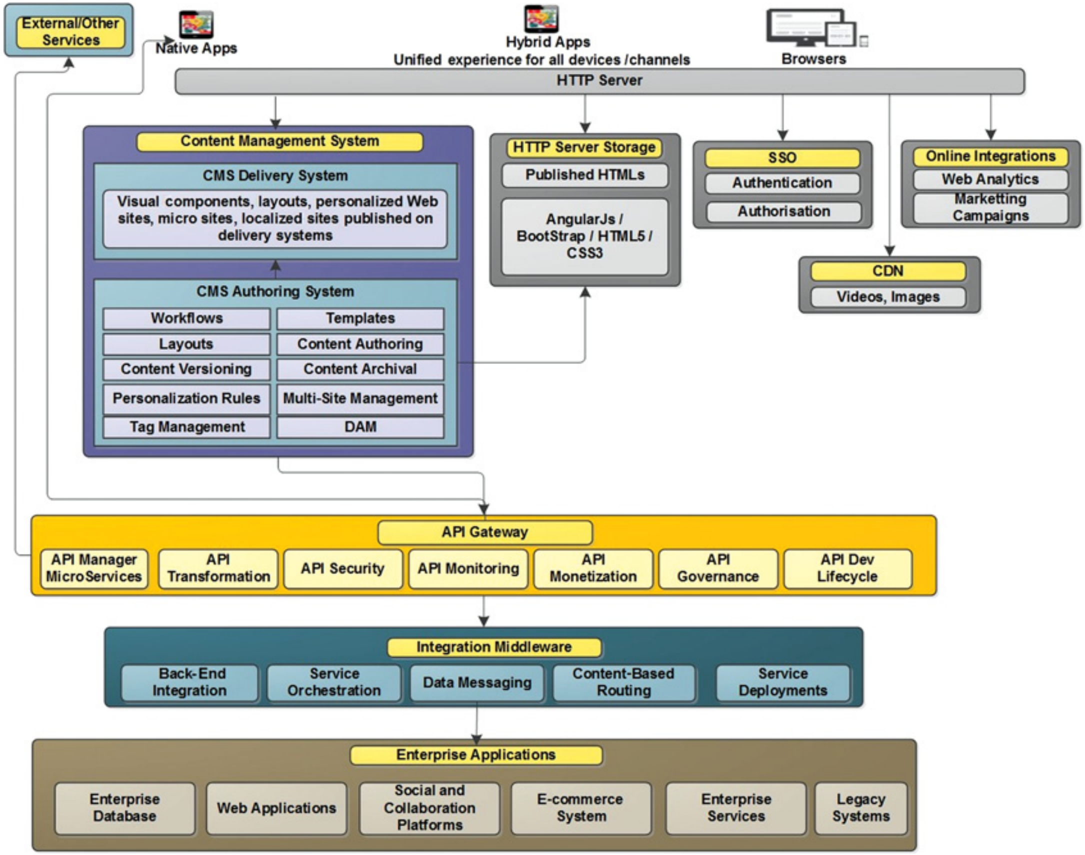


Figure 7.2 Enterprise CMS Integration: The Big Picture

7.3 CMS INTEGRATIONS

In this section we take a look at the integration details of all key CMS integrations. For each integration we will also look at the common challenges and best practices to address those challenges.

Security Integration

A CMS platform provides built-in security features. Let us look at the general aspects of CMS security-related integrations:

- **Authentication**: The main responsibility of this module is to validate user credentials. CMS broadly provides four kinds of authentication mechanism:
  - Internal password-based authentication. CMS stores user authentication details in its internal repository and uses it for authenticating the user. In this scenario the user details are regularly synched from the master user repository (such as corporate LDAP). User provisioning systems or custom synchronization jobs can be used for synchronizing user details. User provisioning systems/custom sync jobs need to understand the underlying data model so that it can properly synchronize the user details.
  - Direct integration with user registries: CMS will be directly integrated with user registries such as LDAP or active directory. Most CMS platforms provide LDAP plugins that can be leveraged for this.
  - Integration with enterprise security managers: In this scenario, there will be enterprise-wide security managers (such as CA SiteMinder, IBM Tivoli Access Manager, Oracle Access Manager), which will authenticate the user and create the user tokens upon successful authentication. CMS will be integrated with security managers for user authentication. Same technique can also be used for providing an SSO solution.
  - Custom security adaptors: Some CMS platforms provide extensible security frameworks (such as JAAS based systems) that can be used to develop security extensions. This will be used in scenarios that have proprietary authentication mechanism or nonstandard user registries.
- **Authorization**: These modules implement fine-grained security controls that specify the user access permissions on specific content resources. CMS platforms provide authorization through robust permission model (allowing administrator to assign user and role permissions to resources), access control lists (ACL), and programmatic APIs. User roles are assigned to resource permissions for implementing authorization. User provisioning systems will be used to synchronize user roles across CMS and enterprise user repository.
- **Single sign-on (SSO)**: CMS platform can be integrated with enterprise-wide SSO. Additionally, various plugin modules for the SSO solutions are available for CMS products. Normally SSO will be implemented using tokens and HTTP headers. OpenID and SAML SSO modules can be used with CMS systems to

provide Web SSO and federated SSO. For instance, CAS module and OAuth connector modules can be used in Drupal for SSO requirements.

Additionally, CMS provides many built-in security-related features such as resource permission model, user role mapping, access control list, and the like. When CMS is integrated with external security systems, we need to perform end-to-end testing to ensure that all built-in features are working as expected.

Translation System Integration

Translation is one of the main requirements of a CMS. Content translation is normally done using a translation workflow. Translation is required when content needs to be published to global sites with different language requirements.

The key steps in the translation management system integration are as follows:

- CMS translation workflow is created to handle the content translation.
- The translation workflow uses the master copy (such as English-language content) as reference for translation. The translation workflow is usually the last step in the content-publishing workflow (translation sometimes is a sub-workflow for the publishing workflow). Sometimes additional approval steps may be needed based on business requirements.
- A translation management system (TMS) is integrated with CMS. Integration is based on APIs or TMS services. Details of integration are given in next section.
- Translated content is used by the publishing workflow to publish content to different geographies.

CMS-TMS Integration

Integration with an external translation management system is normally done through services. Some TMS also offer API-based integration and provides libraries for the same.

Prerequisites for the translation workflow

- The content chunks in the master language (usually in English) should be in "ready to be published" stage. The chunks should be reviewed and approved by all concerned stakeholders.
- CMS should be integrated with the translation management system (through services-based or API-based integration).
- The callback service should be implemented on the CMS. A callback service running on the CMS server ensures proper synchronization between translation workflow (running in CMS) and translation service (running in the TMS server). Callback service will restart the translation workflow by providing the path of the translated content chunks. The callback service will be invoked

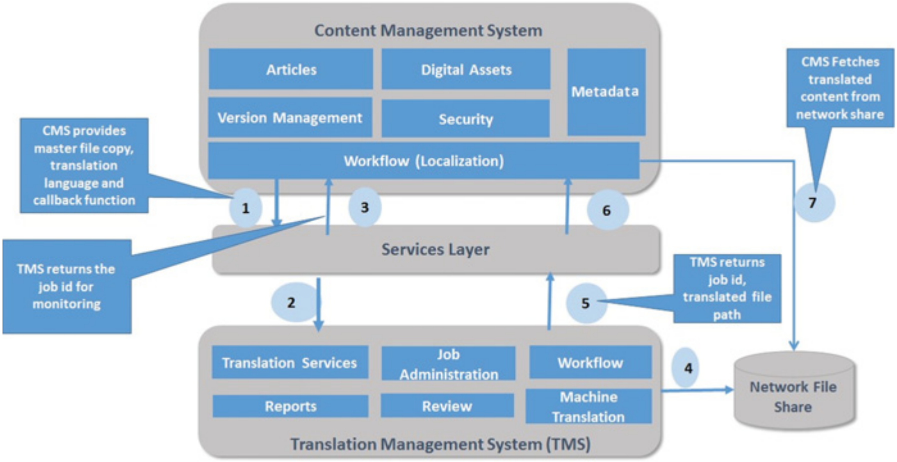

Content Management System
Articles
Digital Assets
Metadata
Version Management
Security
Workflow (Localization)
CMS Fetches translated content from network share
CMS provides master file copy, translation language and callback function
1
3
6
7
Services Layer
TMS returns the job id for monitoring
2
5
TMS returns job id, translated file path
Translation Services
Job Administration
Workflow
4
Network File Share
Reports
Review
Machine Translation
Translation Management System (TMS)

Figure 7.3 CMS-TMS Integration Process

by the TMS once the translation is complete. The callback service can be implemented as REST-based listeners.

The general steps of the translation workflow are given in Figure 7.3.

- In the first step, the translation workflow submits the master content chunk (normally in English) as a job to the TMS system along with its metadata. The content ID and content path will be sent as part of content and will be submitted to the content queue. The submission happens through the invocation of a translation service in second step. The translation service requires key parameters such as translation language, callback service, publishing path, etc. The callback service will be invoked by TMS once the translation is complete. The callback service will be implemented as a service listener on the CMS.
- In third step, the TMS service provides a job ID as the return value. The translation workflow need not wait for the translation to be completed. It can store the job ID mapped with the content ID along with language in a content repository for reference. At this stage the translation workflow will be paused.
- TMS could use manual translation or automated translation to translate the content. In step 5 and 6, when the translation is completed, the TMS will invoke the callback service running on CMS providing the key details such as job ID, path for translated content, etc. In seventh step, The callback function fetches translated content from the location and restarts the workflow.
- Translation workflow then publishes the localized article and notifies appropriate stakeholders.

Translation best practices

- The main challenge in the translation is the performance of the translation process. The end-to-end process would take anywhere between few hours and days based on the capability (such as automated translation, manual translation, support for batch jobs, etc.) of the TMS. Hence the TMS should be evaluated for the solution requirements.
- In order to address the performance issue, it is recommended to use a TMS with batch translation support and ability to handle prioritized translation. Services-based integration and asynchronous content submission are recommended.
- Another common challenge is the ability to support multiple languages. This again largely depends on the capabilities of TMS. The capabilities of TMS should be tested for all supported languages including the double-byte character set languages such as Chinese.
- For optimal performance, it is better to implement the translation as an asynchronous process. CMS submits the content to TMS and pauses the workflow. The process should not block other activities of the workflow. The workflow will be notified by TMS once the translation is complete.

Search Engine Integration

Content search is a key CMS capability to show relevant content based on search keywords and metadata. There are mainly two aspects of CMS and search integration: content search and asset search. Integration between search engine and CMS happens mainly in three forms:

- **Content services and APIs**: A search engine uses the content services or APIs exposed by the CMS to index CMS content. The services and API may provide the links to published content so that the search engine can index published content.
- **Search connectors**: Few search engines provide built-in connectors and adaptors to the CMS. They will directly connect to the content repository and index content. In this scenario the connectors/adaptors need to understand the underlying CMS content model and content repository structure, and to index the content.
- **Search plugins for CMS**: Another popular way to integrate search with CMS is to use CMS search plugins. Search plugins provide seamless search integration for indexing content for a particular CMS. For instance, *Solr plugin for WordPress* provides site-indexing features, faceting on fields (such as categories, tags, page type, custom fields), sorting, spelling suggestions, auto-complete suggestions, and the like. *Apache Solr Search module* for Drupal provides features such as faceted search, content block suggestions, and others.

Given below are some of the points to consider while crawling CMS content:

- **Well-formed content URL:** When a search engine directly connects to CMS and indexes its content, it is important to create a proper URL for content. Few CMS platforms store the URL or path as one of the content properties or as metadata value, and a search engine can then use it to construct a valid public URL.
- **Security:** Though a search engine can index all CMS content, it should filter the search results based on access permissions. CMS platforms normally specify the access list for each part of content (through content properties or through security metadata), and a search engine can use this information for security filtering.
- **Site search through Web crawling:** The aforementioned methods allow the search engine to index content directly from CMS. For a Web site that is predominantly based on CMS content, we can implement the site search feature through Web URL crawling. A search engine will directly crawl and index site content from the starting URL (such as home page or landing page). In this method, the search engine can index published content with clean URLs.

Content Services Integration

Service-oriented architecture enables us to produce and consume modular, reusable services. In such architecture enterprise systems interact with each other via services. In a simple scenario we can use point-to-point services where the consuming system directly invokes the producer service, whereas in a complex scenario (with a large number of systems and services requiring governance and management) we will use message-oriented middleware (MoM) such as enterprise service bus (ESB).

CMS being an important integral part of the enterprise capability plays an active role in service-oriented architecture. CMS exposes the key content functions (content query, content updates, etc.) as external services and consumes services from taxonomy, metadata, and similar systems. CMS will also publish the services similar to the way it publishes content.

A sample publishing service exposed by CMS is shown in Figure 7.4.

Content management systems expose content through services. Figure 7.4 depicts publishing services exposed by CMS used for knowledge management system. Popular services formats are XML-based SOAP Web service and JSON-based REST service. In Figure 7.4, CMS publisher exposes SOAP-based Web service for search systems, REST-based article publishing in JSON format for external content service aggregation layer, and asset services for external asset server.

Normally SOAP-based Web service is used for internal or B2B systems. As XML is the data exchange format, it would be relatively “heavier” in terms of data size. For Web applications, it is recommended to use a “lighter” JSON-based REST service.

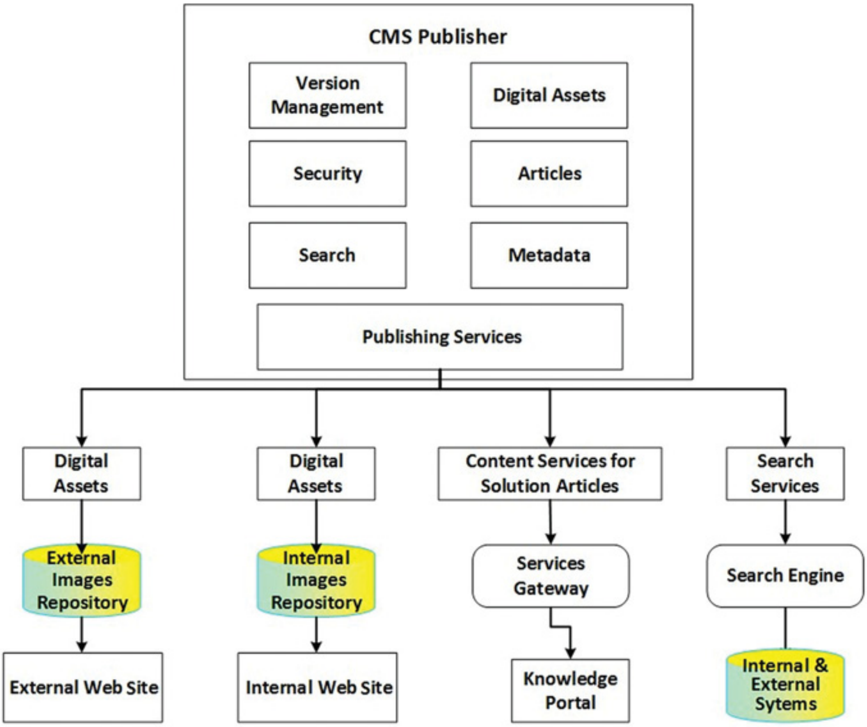


Figure 7.4 CMS Publishing Services

We could also create micro services for various content functions. Through micro-services, we can expose granular content functions to consumers. Micro-services could internally use one or more content services. We had discussed micro services in chapter 4.

Content service best practices

- For designing scalable and high-performance content services that are mainly consumed over HTTP, give preference to REST with JSON over SOAP with XML.
- For long-running content transactions (such as content translation), always use non-blocking asynchronous service call to a service provider with a callback service. A service provider can invoke callback service to notify about submitted request status (it can provide the path to translated content).
- If the presentation view needs to do multiple service calls, design a composite resource (which is an aggregation of all needed resources) to provide all required content with a single service call. Combining content through a composite resource would reduce server round trips and increase performance. For instance, if the products page needs three service calls to get product features,

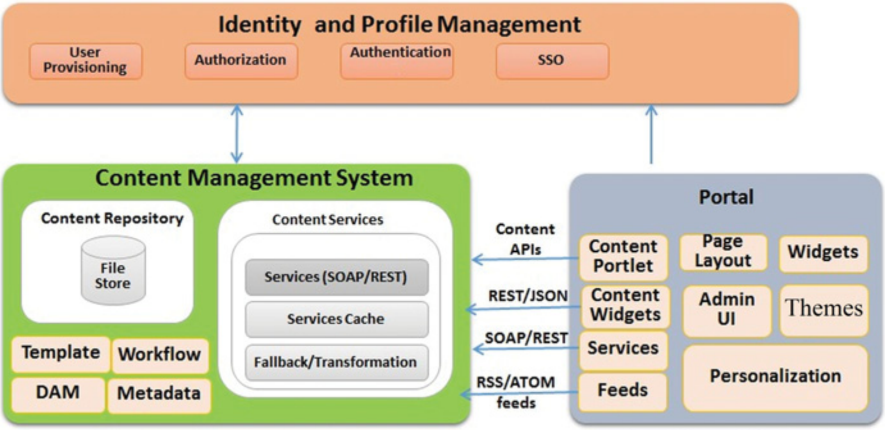


Figure 7.5 CMS Portal Integration

product discussions, and product programs, create a composite resource containing product features, product discussions, and product programs in a single JSON/XML file.

- Add appropriate caching headers (Cache-Control, Expires, Last-Modified, etc.) for the response. We can set higher cache value for static resources such as images, videos, and other multimedia assets. Similarly, the cache headers for static content and error pages can be appropriately set to enhance performance.

Portal Integration

Portals are popular presentation engines for an enterprise that provides personalized user experience by aggregating content from multiple content sources. A portal relies heavily on CMS for Web content management. Enterprise information portals (EIP) get a majority of their content from CMS. The predominant use case is to fetch content based on metadata and keywords.

There are several methods for integrating CMS with portals, as depicted in Figure 7.5.

- **Content portlets**: Most portal platforms offer built-in content portlets that work seamlessly with their native CMS; for instance, the Liferay Portal provides the Web content display portlet, asset publisher portlet, and Web content management portlet to interact with Liferay CMS. We can configure or extend those portlets to connect to other CMS systems. In this approach we use content APIs or services exposed by CMS.
- **CMS services**: We can build custom connectors and portlets using CMS content services. For external-facing Web portals, we generally use REST services.

- **Content APIs**: We can build custom connectors and portlets using content APIs. If the CMS is JCR compliant, we can use JCR APIs from portal to fetch CMS content.
- **Content widgets**: Widgets are lightweight components built using JavaScript and similar client-side technologies. Widgets can fetch content asynchronously from CMS, and internally they will use content services or content APIs.
- **Feed portlet**: A portal can also consume the content updates through content feeds in RSS/ATOM formats.

Note: In modern portals and CMS systems, there is increasing convergence between presentation and content management concerns. Portals are coming bundled with lightweight content management systems, and CMS are offering lightweight portal presentation platforms. The integration techniques discussed above assume CMS integration with a full-fledged horizontal portal.

Presentation Engine Integration

Apart from portals, a custom presentation engine can also be built using Web technologies such as HTML 5 along with JavaScript frameworks. In such scenarios, the presentation engines will mainly use content services to fetch CMS content. Alternatively, CMS can publish content in the form of HTML, XHTML, or XML to the Web server, and the presentation engine can use published content for rendition.

Metadata Management System (MMS) Integration

Most CMS provide built-in metadata and tagging/annotation support to add semantic metatags for content and assets. This may be useful in most of the cases. In some enterprises we may see enterprise-wide taxonomy systems that are used by multiple enterprise applications. These taxonomy systems store the terms, vocabulary, and definitions for terms applicable across entire enterprise applications. In such cases we will integrate CMS with enterprise metadata management systems.

The following sections introduce us to some of the methods for integrating CMS with MMS.

API and service-based integration

CMS with be integrated with MMS systems through exposed services and APIs. We can use the extensible CMS plugin architecture to develop MMS integrators using APIs and services. For performance reasons, we will synchronize the taxonomy terms with CMS on regular basis and store a local copy of those terms on CMS.

JavaScript-based integration

Few MMS platforms (such as Synaptica) also provide tagging Web interface and JavaScript-based integration. High-level integration steps in such cases are as follows:

- We will configure the taxonomy Web URL in the CMS authoring templates (for the metadata field).
- Before invoking the taxonomy URL we will place certain form variables on the page. These will be hidden fields used for communication with MMS.
- A JavaScript function embedded within a CMS page invokes the taxonomy Web URL by passing the MMS variable values. MMS renders a tagging interface that can be used for tagging.

Feed Integration

Feeds are another effective way to distribute content. While services expose modular content functions for consuming applications, feeds expose content updates, events, calendar updates, content notifications, popular content, trending content, and similar items for consuming applications.

RSS and ATOM are popular XML-based feed formats. Most of the CMS platforms provide plugin modules to expose content through feeds and to consume external feed content. For instance, *Views Atom* and *Feeds Atom* modules can be used for processing feeds in Drupal, and WordPress has built-in support to expose content through RSS/ATOM feeds. In Joomla we can use RSS Feed Reader extension for publishing content as feed.

Digital Asset Management (DAM) Integration

Digital asset management (DAM) is one of the vital components in managing content. DAM stores the static assets such as images, videos, graphics, media files, flash files, documents, binary files, etc. We will explore the DAM capabilities in more detail in Chapter 8.

In most cases, we need to use these assets while authoring or editing content. While many CMS provide built-in support for managing digital assets, sometimes these need to be integrated with enterprise DAM systems.

There are two integration touch points between CMS and DAM systems:

- Synchronizing a CMS asset repository with an enterprise DAM system
- Integration between CMS and DAM for all asset needs

The first option is used when CMS has its own asset repository. In this case, a regular synch job will be developed to synchronize assets between CMS asset repository and the DAM system. The synchronization job should also synchronize the asset metadata and various renditions of the asset (such as various image resolutions).

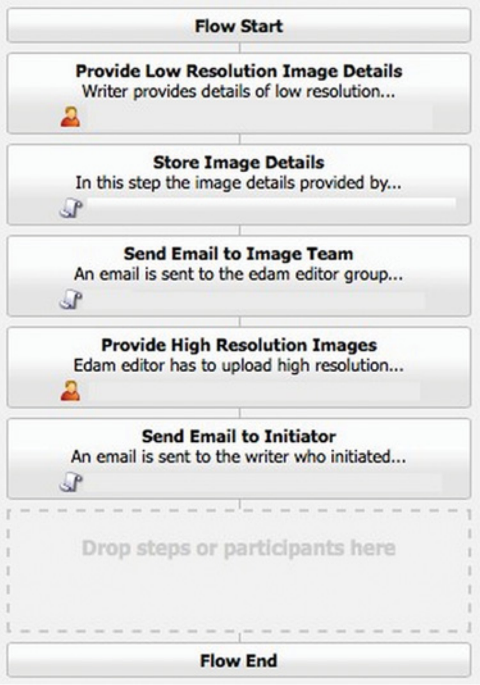

Flow Start
Provide Low Resolution Image Details
Writer provides details of low resolution...
Store Image Details
In this step the image details provided by...
Send Email to Image Team
An email is sent to the edam editor group...
Provide High Resolution Images
Edam editor has to upload high resolution...
Send Email to Initiator
An email is sent to the writer who initiated...
Drop steps or participants here
Flow End

Figure 7.6 Image Request Workflow

We will use a second option when the DAM owns all the asset management functionality. CMS will get all the assets from DAM in real time while adding image/media assets to content. This will be usually implemented through CMS asset workflows. Assets will be uploaded to the enterprise DAM through "asset upload workflow," and assets will be requested through the "asset request workflow." Usually DAM provides various renditions (such as various sized images) of the asset. We can request for the appropriate asset rendition in the asset request workflow. In this case, DAM will be responsible for maintaining asset metadata tags and asset searching.

In both cases, when content is ready for publishing, the publishing workflow will ensure that all associated images (along with its metadata) are also published to the target location.

A simple asset workflow is depicted in Figure 7.6.

The workflow is initiated within CMS when the content author requests a high-resolution version for an existing low resolution image. This triggers a notification to the Enteprise DAM (EDAM) editor group to provide the high-resolution version of the specified image. Once the image of proper resolution is uploaded by the EDAM editor, the author is notified via -mail.

JCR-Based Integration

Content Repository API for Java (JCR) provides a platform-independent standardized APIs for accessing content from compliant content repositories. JCR standards are defined as part of JSR 170 (JCR 1.0) and JSR 283 (JCR 2.0). JCR 1.0 provides features such as versioning, queries search, events, persistent transactions, content type system, and content structure, and JCR 2.0 focuses on shareable nodes and SQL query extensions.

JCR abstracts the caller from internal details of the target repository (such as DBMS, CMS or file system) and the underlying data structure details (such as hierarchical data, structured or semi-structured data). JCR offers various content services for content updates, content reads, content search, versioning, and others.

JCR-compliant content repository is made up of multiple workspaces. Each workspace contains a hierarchy of "items," and each item can be a "node" or a "property." It stores the information in a node-based tree structure. Each node represents content along with properties (such as content type, content path, etc.). We can use the interfaces from the javax.jcr package for accessing the compliant content repository. Apache JackRabbit is a reference implementation of JSR 170. We have discussed the details of JCR along with a code example in Chapters 3 and 4.

JSR specification outlines various access levels (level 1 and level 2) for the content repository. We can use the JCR APIs to retrieve content from JCR-compliant enterprise content repositories. Migrating across JCR-compliant repositories is also easier using XML-based data exchange.

7.4 CMIS-BASED INTEGRATION

Content management interoperability services (CMIS) standard provides easier interoperability across compliant enterprise content platforms. A CMIS standard-compliant system exposes language-independent and platform-independent standard services via SOAP-based Web services and REST-based AtomPub protocol. CMIS provides CRUD (Create, Read, Update, and Delete) operations on top of a compliant content repository.

A CMIS domain module mainly consists of four types of objects:

- Folder objects representing the document containers for a given content repository
- Document objects representing content entities
- Relationships specifying the relationship between two objects
- Policy objects that can be applied to other objects

CMIS services include services such as discovery services (for executing CMIS query), object services (to get details about the object), relationship services (to get object relationships), and the like. Clients can use them to perform content operations in a CMIS repository. CMIS offers service layer abstraction for client applications from an underlying repository. Various repositories such as database, CMS, ECM, and file systems can be abstracted by CMIS. Client applications can use services to access various repositories through CMIS. CMIS implementation abstracts the client

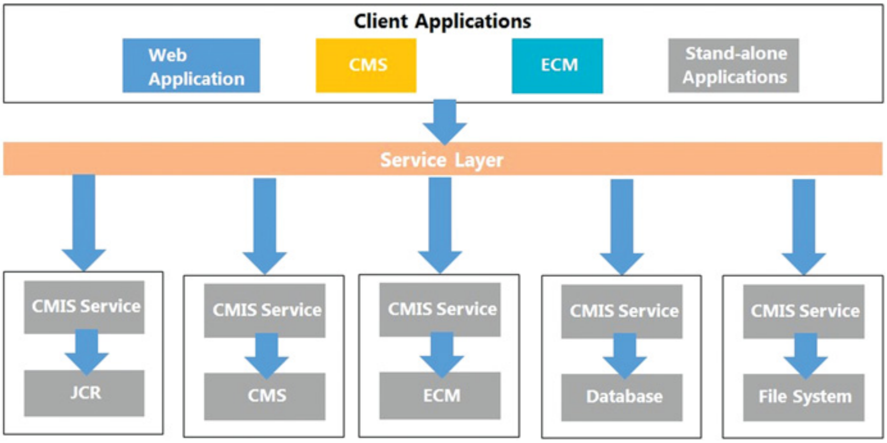


Figure 7.7 CMIS-Based Integration

from underlying system details and provides a consistent interface via services. The CMIS-based integration setup is depicted in Figure 7.7.

When can we use CMIS-based integration?

- When there are multiple content repositories in an enterprise (which includes CMIS-compliant ECM content repositories) and we want to publish content from one content repository to another.
- When we want to integrate WCM and CMIS-compliant content management systems
- When we want to get an integrated view of content from multiple enterprise content repositories
- When we want to do a federated search across content repositories

For instance, for integration between Drupal and Alfresco (which is a fully compliant CMIS system), we can leverage the *CMIS API package of modules*. Using these modules we can create, update, and search for Alfresco ECM content via Drupal interface. The CMIS module package provides CMIS client API, CMIS repository browser, CMIS query module, and CMIS sync module.

JCR and CMIS

Both JCR and CMIS are prominent content standards that can be used for seamless content interoperability to build composite applications. Table 7.1 provides general usability scenarios where we can adopt JCR and CMIS.

Table 7.1 JCR and CMIS usage scenarios

	JCR	CMIS
Access mechanism	Mainly through APIs from javax.jcr package (implementation available for other languages as well) HTTP access through Apache Sling	Mainly through SOAP and REST services
Reference implementation	Apache JackRabbit	Apache Chemistry
Interoperability	JCR can work with a wide variety of content such as Web content, digital assets, documents, etc.	CMIS mainly focuses on file and document management
Usability scenario	When application wants to access content through API or HTTP interface from compliant repositories	When service-based access is required from compliant repositories

7.5 CMS INTEGRATION WITH OTHER SYSTEMS

Based on business needs, CMS needs to be integrated with other systems such as social and collaboration systems, enterprise content applications, campaign management application, Web analytics application, forms-based applications, e-commerce platforms, knowledge management applications, cloud-based applications, enterprise databases, ESB, and the like. Table 7.2 provides generic integration methods for integrating with these systems.

So far we have seen various CMS integration methodologies. In the next section we look at industry standard and effective content standards that can be used for sharing, publishing, syndicating, and integrating content.

7.6 CONTENT STANDARDS

Various content formats are used while creating, storing, publishing, and sharing content. Content based on open standards also ease the job of content migration. It is always recommended to use an open standard-based content format during such content operations. This enhances interoperability, reusability, and extensibility for content. In this section we look at various standards in detail.

The main goals of content standards are:

- To provide a clean separation of core content from its end-presentation and formatting concerns
- To make content easier to exchange and share with other systems and services
- To ease the content localization effort through effective integration with translation management systems

Table 7.2 CMS Integration Methods

System	Integration Method
Collaboration System (wiki, blog, community)	• Feed-based integration (CMS can get content feeds from a collaboration platform) • Services-based integration
External Social Platforms	• OAuth-based integration • API-based integration
Enterprise Content Systems (other CMS and ECM platforms)	• CMIS-based integration (based on compatibility with CMIS) • JCR-based integration (based on compatibility with JCR) • Content services–based integration • API-based integration • Content migration (for one-time content consolidation) • Content synchronization (using frequent sync jobs)
Campaign Management System	• API-based integration • JavaScript-tag-based integration
Analytics/BI Applications	• Content services–based integration • Content API-based integration
E-commerce Applications	• Content services–based integration • Content API-based integration
Knowledge Management Application	• Content services–based integration • Content API-based integration
ESB	• Content services–based integration

- To enhance the productivity of authors and publishers and to reduce the overall turnaround time for content publishing
- Allow for future extensions and modifications
- Maximize the reusability of content
- Make content and its dependencies (metadata, assets) compatible and interoperable with other systems.

Key content standards used for content creation and exchange are given below. They can be broadly classified into two groups: *content publishing/exchange formats* and *content service formats*.

Content publishing and exchange formats include HTML/XHTML, XML, DITA, JSON, and SCORM that are widely used for publishing content into a variety of delivery platforms. As they are open standards, they are also used to create interoperable content, which can be easily exchanged across compatible systems.

CMS exposes content as content services that can be consumed by other systems. In this section we review feed formats (RSS/ATOM) and Web services (SOAP and REST). We begin with content-publishing formats.

HTML/XHTML

This is one of the most popular formats for content publishing. It is recommended that CMS and content services publish core content without presentation logic. This would help the presentation engines and consuming applications (such as portals) transform content to suit their needs.

In some cases, however, the main presentation platform is Web and in such cases CMS can directly publish content in HTML/XHTML format to the Web servers. To support this kind of publishing, CMS should support a presentation template or a content transformation engine, which injects presentation code (such as styles, format, and other visual aspects) into core content. Authors can also use rich text editors (RTE) that can add HTML styles to the content in this scenario. Authors can use presentation templates to specify the style for their content elements. Presentation templates are designed to render content optimized for specific delivery platforms such as Web or mobile device. We can apply presentation templates to optimize the content for various delivery platforms. CMS portals also use HTML format for content publishing.

XSL (Extensible Stylesheet Language) is another powerful content transformation technology. Core content in XML format can be transformed into HTML with an XSL-based transformation engine.

XML (Extensible Markup Language)

This is the most popular general-purpose content format used widely for various content exchanges and sharing scenarios. It is a platform-independent, open standard supported by a wide variety of tools. We can specify the element attributes, element hierarchy, topic definition, step sequence, and other elements using it. We can store only core content in XML format and leverage tools such as XSL for content transformation. The salient features of content in XML format are as follows:

- Representing the content in XML format will also help us in creating “content chunks” that can be reused across various contexts.
- We can also validate the XML content against an XML schema or DTD.
- Using the XML structure, we can specify the content structure, content types, content metadata, ownership and permission, content version, content lifecycle stages, and other similar content attributes.
- Content represented in XML format makes it easily discoverable.
- Maximizes interoperability across various systems and services, as XML is one of the most widely used and supported data formats for information exchange.

- Enables "topic-based authoring" to create modular content that can be reused for multiple contexts and helps in dynamic assembly of personalized content chunks.
- Enables "single-source publishing" through which content in XML format can be published into a variety of formats.
- XML-based content makes Omni-channel publishing easier.

Role of DTD (Document Type Definition) and XSD (XML Schema Definition) in XML content

In order to provide structure and specification grammar for the content in XML format, we can use DTD or XSL. This allows us to enforce the validations, restrictions, and other specifications such as definitions, element ordering, allowed element attributes, element relationships, mandatory property, predefined attribute values, and other elements.

XML-based content publishing

Publishing XML-based content to delivery channels essentially consists of three steps:

- Content authoring and validation: The authoring interface should enforce the rules specified by the DTD or XML schema.
- Content storage: Once the authoring is completed, the content should be saved and stored in predefined XML format.
- Content transformation and presentation: An appropriate transformation engine (such as XSL-based transformation) should be used to transform the XML content into a required rendition type (such as HTML).

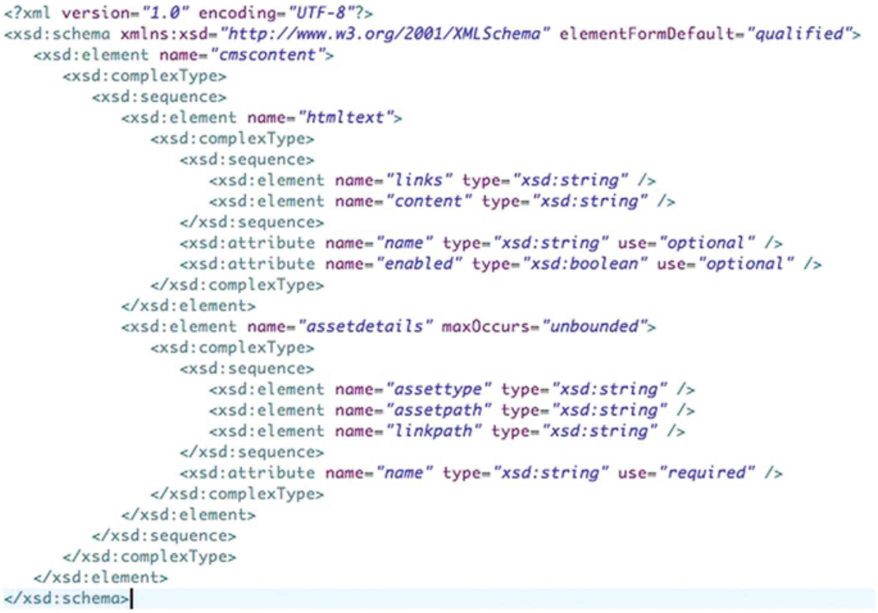
Sample XSL and content XML

Most of the CMS tools provide support for storing and exporting content in the form of XML. We can define a schema in XSD and use it to structure the content XML. A sample XSD and its content XML follows.

Content.xsd The content.xsd specifies the following, as shown in Figure 7.8:

- The structure and element sequence of the "cmscontent" element, "htmltext" element, and "asset" element
- Data type for all the elements within the "cmscontent" element
- It specifies "assetdetails" element as a repeatable element (using maxoccurs = "unbounded")

Content XML The sample content XML shown in Figure 7.9 satisfies the specified schema in content.xsd.



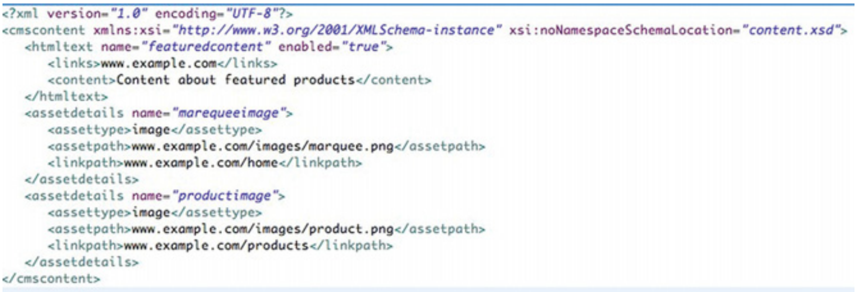
```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
   <xsd:element name="cmscontent">
      <xsd:complexType>
         <xsd:sequence>
            <xsd:element name="htmltext">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="links" type="xsd:string" />
                     <xsd:element name="content" type="xsd:string" />
                  </xsd:sequence>
                  <xsd:attribute name="name" type="xsd:string" use="optional" />
                  <xsd:attribute name="enabled" type="xsd:boolean" use="optional" />
               </xsd:complexType>
            </xsd:element>
            <xsd:element name="assetdetails" maxOccurs="unbounded">
               <xsd:complexType>
                  <xsd:sequence>
                     <xsd:element name="assettype" type="xsd:string" />
                     <xsd:element name="assetpath" type="xsd:string" />
                     <xsd:element name="linkpath" type="xsd:string" />
                  </xsd:sequence>
                  <xsd:attribute name="name" type="xsd:string" use="required" />
               </xsd:complexType>
            </xsd:element>
         </xsd:sequence>
      </xsd:complexType>
   </xsd:element>
</xsd:schema>
```

Figure 7.8 Sample Content XSD

Structured publishing with XML and open standards

Various elements of XML-based structured publishing are shown in Figure 7.10. The four key elements of structured publishing are:

- Content document represented in structured XML format
- Structured rules, which can be specified through DTD or XML schema for the document
- Presentation rules (such as visual format, stylesheets, JavaScripts) that can be configured in XSL
- Final presentation in HTML, XHTML, JSON, or PDF format. Applying the presentation rules on top of the XML document can render this final presentation.



```
<?xml version="1.0" encoding="UTF-8"?>
<cmscontent xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:noNamespaceSchemaLocation="content.xsd">
   <htmltext name="featuredcontent" enabled="true">
      <links>www.example.com</links>
      <content>Content about featured products</content>
   </htmltext>
   <assetdetails name="marequeeimage">
      <assettype>image</assettype>
      <assetpath>www.example.com/images/marquee.png</assetpath>
      <linkpath>www.example.com/home</linkpath>
   </assetdetails>
   <assetdetails name="productimage">
      <assettype>image</assettype>
      <assetpath>www.example.com/images/product.png</assetpath>
      <linkpath>www.example.com/products</linkpath>
   </assetdetails>
</cmscontent>
```

Figure 7.9 Content XML

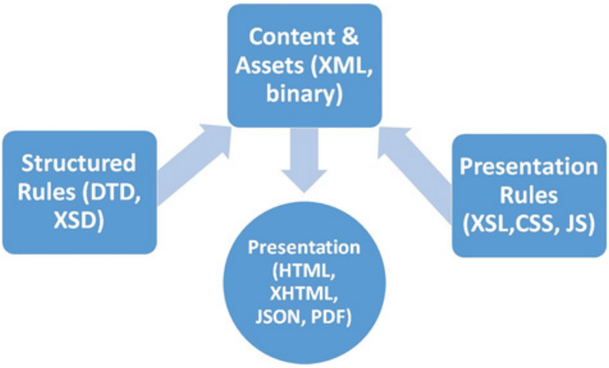

Content & Assets (XML, binary)
Structured Rules (DTD, XSD)
Presentation Rules (XSL,CSS, JS)
Presentation (HTML, XHTML, JSON, PDF)

Figure 7.10 XML-Based Structured Publishing

DITA (Darwin Information Typing Architecture)

DITA is an XML-based architecture used for content creation and publishing. We can specify the content structure and schema in DTD format. The standard is mainly used for easier content creation and efficient sharing and content reuse. We can create DITA topics, DITA maps, DITA composites, schemas, and style sheets.

The main elements of DITA are:

- DITA maps: They help in organizing content by maintaining references to DITA topics and its hierarchies.
- DITA base object: It provides a high-level container for topics and help in creation of the final XML document.
- DITA topics: It refers to a logic unit of content. DITA supports mainly three types of topics: DITA concept topic (specialized topic that usually contains background information), DITA reference topic (specialized topic that provides quick links to information), and DITA task topic (specialized topic that provides how-to information in step-by-step information format).
- DITA content reference: It is the main enabler of content reusability wherein a DITA topic can refer to a reusable content chunk. This allows us to assemble a larger content piece through references of smaller, reusable content chunks.

The key advantages of using DITA in content exchange and content publishing are as follows:

- The major advantage is that it provides a structured reusable format for content exchange. It creates structured content, which adheres to the specified DTD rules. DITA topics (through <topicref>), elements, (through <conref>), and maps (through <anchorref>) can be reused across various scenarios, and DITA supports conditional reuse as well.
- DITA provides topic orientation by allowing us to author content related to a unique subject as part of a single topic.

- DITA format helps in publishing to various presentation formats such as HTML and PDF.
- DITA supports built-in filters, which can help in conditional content publishing by enhancing content reusability.
- DITA supports many other powerful features such as inheritance, specialization, metadata-aware content, conditional filtering, content reusability, Omnichannel publishing, translation support, and modularity, which comes handy in content authoring and publishing.
- DITA content optimizes translation and authoring effort, bringing in good return on investment.
- Metadata support in DITA helps in content discovery and content search.
- DITA helps in Web page creation through aggregation of smaller, modular content chunks. It is possible to combine DITA topics with other related topics to create the appropriate content set.

Most CMS support DITA format. Drupal has DITA integration module; DITA4Joomla can be leveraged to author and manage DITA content within Joomla CMS; DITA content can be published from WordPress.

Authoring and publishing with DITA

To create and manage DITA content we need to use DITA-compliant authoring tools within CMS. DITA Open Toolkit is one of the popular DITA tools, and many CMS provides seamless support to DITA toolkit. DITA toolkit ensures that authored content is DITA-compliant and checks that content is well formed. It also validates tag structure, hierarchy, and nesting against specified DTD rules. DITA Open Toolkit can also be used for publishing DITA content in various formats such as HTML, PDF, HTML, and others.

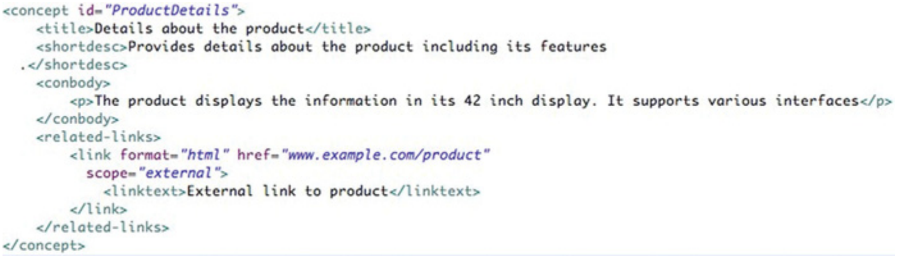
Using DITA for creating various content types

We can leverage DITA elements to author content for various scenarios. A sample mapping of DITA elements to the content scenarios is given below:

- Concept topic: This can be used to provide a detailed description of a subject matter. It contains attributes and related links of the subject matter.
- Task: We can use it to describe the detailed steps. This would be handy in authoring “How-to” documents providing step-by-step instruction.
- Topic map: We can specify metadata (such as author, publisher, date, timestamp, etc.) and refer it to topic documents.

A sample DITA concept topic for describing a product is shown in Figure 7.11.

A sample task topic providing step-wise instructions for a solution article in DITA format is shown in Figure 7.12:



```
<concept id="ProductDetails">
    <title>Details about the product</title>
    <shortdesc>Provides details about the product including its features
  .</shortdesc>
    <conbody>
        <p>The product displays the information in its 42 inch display. It supports various interfaces</p>
    </conbody>
    <related-links>
        <link format="html" href="www.example.com/product"
          scope="external">
            <linktext>External link to product</linktext>
        </link>
    </related-links>
</concept>
```

Figure 7.11 Sample DITA Concept Topic

JSON (JavaScript Object Notation)

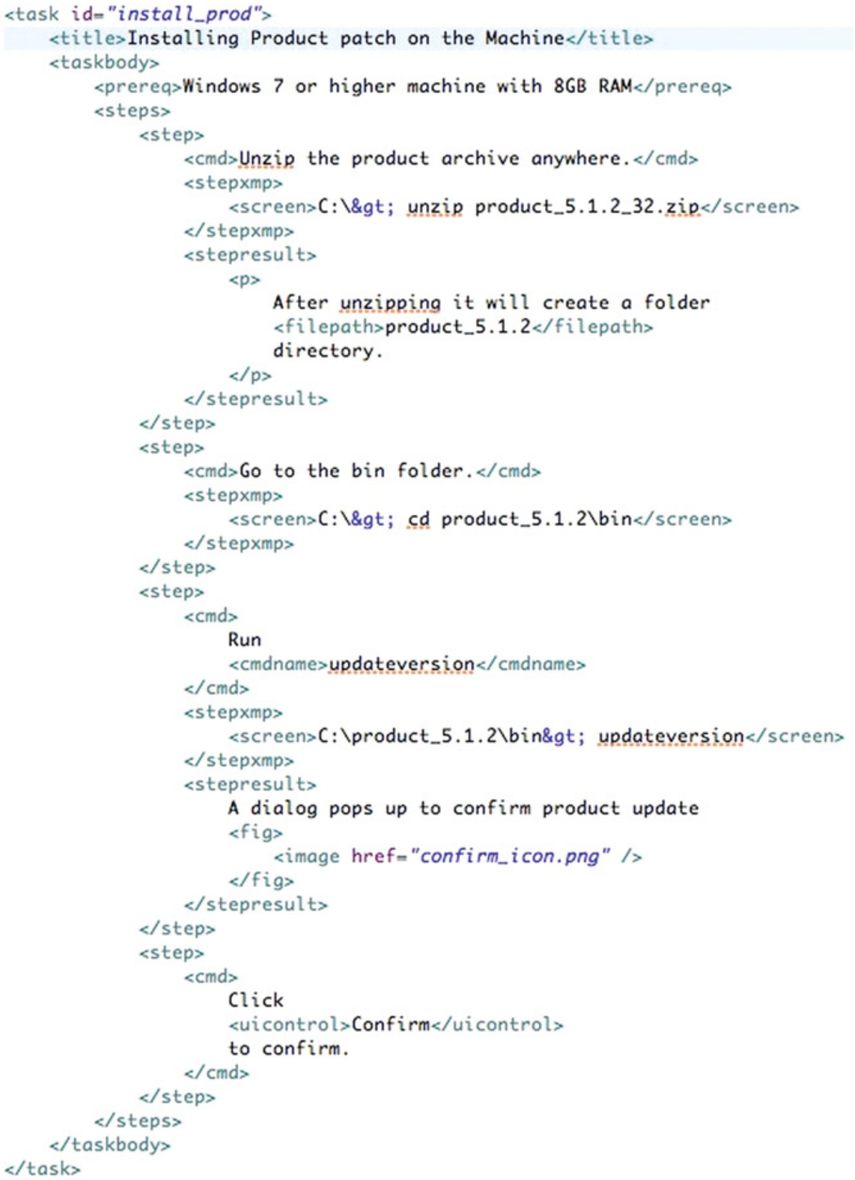
JSON is one of the most popular lightweight data exchange formats for the Web. Most of the JavaScript libraries and browsers provide built-in supporting for JSON parsing and rendering. As it is lightweight and flexible compared to other formats, mobile-enabled Web sites and high-transaction Web sites prefer JSON for exchanging content. Here are some of the other applications of JSON-based content:

- Omni-channel readiness: CMS can provide content to mobile-enabled systems and mobile Web in JSON format.
- High scalability: Content represented as JSON achieves high scalability.
- Integration with internal and external applications: CMS can also expose content as JSON other internal and external systems, which can consume JSON-based content.
- REST-based content APIs: CMS can use REST-based APIs to expose various CMS operations to external systems. APIs for content node operations, content query operations, describing content object model, and the like can use REST APIs using JSON data format.

A sample home page content representation in JSON format is shown in Figure 7.13. It shows various sections that are part of a home page and content within those sections.

SCORM (Sharable Content Object Reference Model)

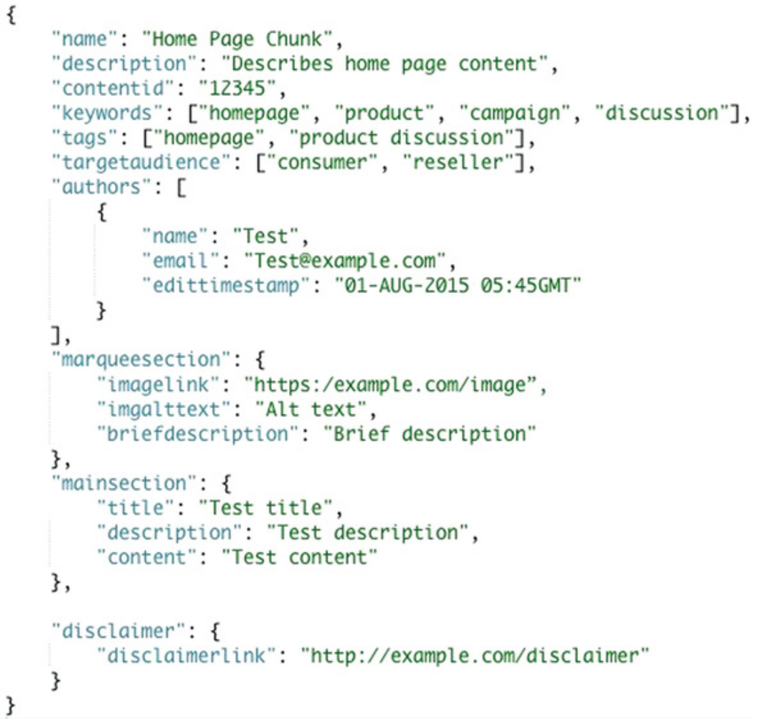
The SCORM standard provides a means to create reusable, discoverable, and shareable learning content. The standard is mainly used for content interoperability among various learning management systems (LMS) and training systems. A SCORM-compliant content can be easily migrated from one LMS system to another. SCORM is based on DITA in the context of learning and training. The salient features of the standard are:



```
<task id="install_prod">
    <title>Installing Product patch on the Machine</title>
    <taskbody>
        <prereq>Windows 7 or higher machine with 8GB RAM</prereq>
        <steps>
            <step>
                <cmd>Unzip the product archive anywhere.</cmd>
                <stepxmp>
                    <screen>C:\&gt; unzip product_5.1.2_32.zip</screen>
                </stepxmp>
                <stepresult>
                    <p>
                        After unzipping it will create a folder
                        <filepath>product_5.1.2</filepath>
                        directory.
                    </p>
                </stepresult>
            </step>
            <step>
                <cmd>Go to the bin folder.</cmd>
                <stepxmp>
                    <screen>C:\&gt; cd product_5.1.2\bin</screen>
                </stepxmp>
            </step>
            <step>
                <cmd>
                    Run
                    <cmdname>updateversion</cmdname>
                </cmd>
                <stepxmp>
                    <screen>C:\product_5.1.2\bin&gt; updateversion</screen>
                </stepxmp>
                <stepresult>
                    A dialog pops up to confirm product update
                    <fig>
                        <image href="confirm_icon.png" />
                    </fig>
                </stepresult>
            </step>
            <step>
                <cmd>
                    Click
                    <uicontrol>Confirm</uicontrol>
                    to confirm.
                </cmd>
            </step>
        </steps>
    </taskbody>
</task>
```

Figure 7.12 Sample DITA Task with Sequence Steps

- The SCORM standard specifies self-contained learning objects using DITA topics and maps. For instance, a DITA map can be used for assembling a logical instructional unit including a course along with its chapters.
- The standard uses DITA metadata to facilitate faster indexing and makes SCORM content easily accessible.



```
{
    "name": "Home Page Chunk",
    "description": "Describes home page content",
    "contentid": "12345",
    "keywords": ["homepage", "product", "campaign", "discussion"],
    "tags": ["homepage", "product discussion"],
    "targetaudience": ["consumer", "reseller"],
    "authors": [
        {
            "name": "Test",
            "email": "Test@example.com",
            "edittimestamp": "01-AUG-2015 05:45GMT"
        }
    ],
    "marqueesection": {
        "imagelink": "https:/example.com/image”,
        "imgalttext": "Alt text",
        "briefdescription": "Brief description"
    },
    "mainsection": {
        "title": "Test title",
        "description": "Test description",
        "content": "Test content"
    },

    "disclaimer": {
        "disclaimerlink": "http://example.com/disclaimer"
    }
}
```

Figure 7.13 Home Page Content in JSON

- It enables easier exchange of content among various learning systems through reusable content objects.
- Enables easier reuse and interoperability of learning materials such as courses, lectures, presentations, and instruction modules.
- The SCORM standard provides a clean separation of core content from its context and presentation.

Comparison of XML, DITA, and SCORM

Table 7.3 provides a general comparison of the content format we have discussed so far:

Feed Formats: RSS/ATOM

Content that gets updated regularly is exposed as feeds. RSS (Really Simple Syndication) and ATOM are two popular feed formats. Many content systems provide default support for exposing content in feed format. News Web sites, blog sites,

Table 7.3 XML, DITA, and SCORM comparison

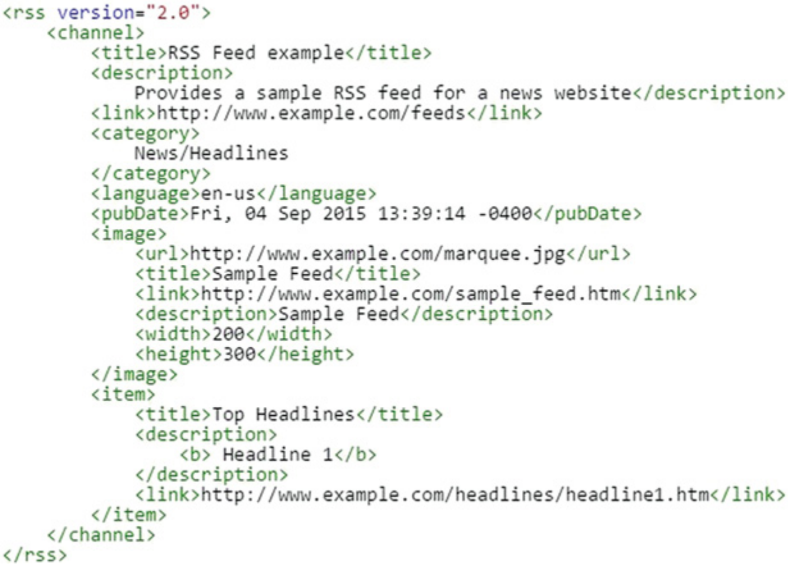
	XML	DITA	SCORM
Primary purpose	Provides structured content Can specify restrictions through XML schema	Provides modular, extensible, structured, and reusable content in the form of topics, concepts, tasks, and references	Provides structured learning items, packaging, and sequencing of learning content
Functional domain	Can be used for any domain along with a specific schema	Specialized DITA for various domains such as publishing	Learning management systems and education domain
Publishing support	Through XSLT and such transformation engines	Supported through DITA open toolkit	Built-in support for publishing
Interoperability	Supported by most CMS systems	Supported by a wide variety of CMS	Highly interoperable among compliant learning systems
Key elements	XML tags and attributes	Topics, maps, tasks, references, links	Learning objects, blocks, curricular
Main advantages	General purpose content exchange format	Enhances reusability, reduces translation effort We could leverage specialization, multi-channel delivery, conditional processing, and filtering Enables single-source content publishing	Enables exchange of learning information

and other similar sites that display frequently updated information (e.g., stock values) may expose content in feed format. Feed formats are generally used to notify the subscribed users of content updates. Either an entire page or specific page portions/sections can be exposed as content feed. Both RSS and ATOM are XML-based formats, which can be used to syndicate content.

A sample RSS feed for publishing news content is shown in Figure 7.14.

Web Service Standards: SOAP and REST

Content systems may expose underlying content through content services. SOAP and REST are the popular Web service standards for exposing content.



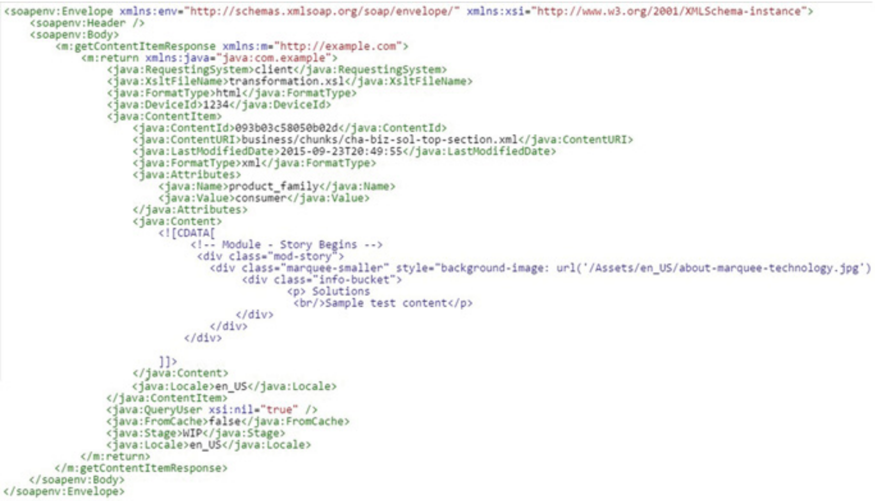
```
<rss version="2.0">
    <channel>
        <title>RSS Feed example</title>
        <description>
            Provides a sample RSS feed for a news website</description>
        <link>http://www.example.com/feeds</link>
        <category>
            News/Headlines
        </category>
        <language>en-us</language>
        <pubDate>Fri, 04 Sep 2015 13:39:14 -0400</pubDate>
        <image>
            <url>http://www.example.com/marquee.jpg</url>
            <title>Sample Feed</title>
            <link>http://www.example.com/sample_feed.htm</link>
            <description>Sample Feed</description>
            <width>200</width>
            <height>300</height>
        </image>
        <item>
            <title>Top Headlines</title>
            <description>
                <b> Headline 1</b>
            </description>
            <link>http://www.example.com/headlines/headline1.htm</link>
        </item>
    </channel>
</rss>
```

Figure 7.14 RSS Format News Content

SOAP

SOAP (Simple Object Access Protocol) is essentially an XML-based service, which uses WSDL document to describe the service details.

Figure 7.15 is a sample of content provided through a SOAP response. We can see that in addition to content (within the <java:content> element), the response contains various attributes listing the content metadata (such as language, content ID,



```
<soapenv:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <soapenv:Header />
    <soapenv:Body>
        <m:getContentItemResponse xmlns:m="http://example.com">
            <m:return xmlns:java="java:com.example">
                <java:RequestingSystem>client</java:RequestingSystem>
                <java:XsltFileName>transformation.xsl</java:XsltFileName>
                <java:FormatType>html</java:FormatType>
                <java:DeviceId>1234</java:DeviceId>
                <java:ContentItem>
                    <java:ContentId>093b03c58050b02d</java:ContentId>
                    <java:ContentURI>business/chunks/cha-biz-sol-top-section.xml</java:ContentURI>
                    <java:LastModifiedDate>2015-09-23T20:49:55</java:LastModifiedDate>
                    <java:FormatType>xml</java:FormatType>
                    <java:Attributes>
                        <java:Name>product_family</java:Name>
                        <java:Value>consumer</java:Value>
                    </java:Attributes>
                    <java:Content>
                        <![CDATA[
                             <!-- Module - Story Begins -->
                              <div class="mod-story">
                                <div class="marquee-smaller" style="background-image: url('/Assets/en_US/about-marquee-technology.jpg')
                                     <div class="info-bucket">
                                            <p> Solutions
                                             <br/>Sample test content</p>
                                    </div>
                                </div>
                            </div>

                        ]]>
                    </java:Content>
                    <java:Locale>en_US</java:Locale>
                </java:ContentItem>
                <java:QueryUser xsi:nil="true" />
                <java:FromCache>false</java:FromCache>
                <java:Stage>WIP</java:Stage>
                <java:Locale>en_US</java:Locale>
            </m:return>
        </m:getContentItemResponse>
    </soapenv:Body>
</soapenv:Envelope>
```

Figure 7.15 Sample SOAP Content

content chunk path, etc.). In this example content is in HTML format, but it can also be served in its original XML format in the SOAP response, and the presentation engine can transform it into HTML based on its needs.

REST

REST (Representational State Transfer) is an architectural style for designing loosely coupled, scalable Web services. It provides a lightweight alternative for exposing Web content and content operations through resource URLs over HTTP. REST is stateless and provides HTTP methods (GET, POST, UPDATE, DELETE) for performing the operations on server side resources. REST is gaining increasing popularity due to its simple yet powerful structure and is adopted by a wide variety of CMS for content operations and for exposing content services.

In a content scenario, REST services mainly serve two purposes:

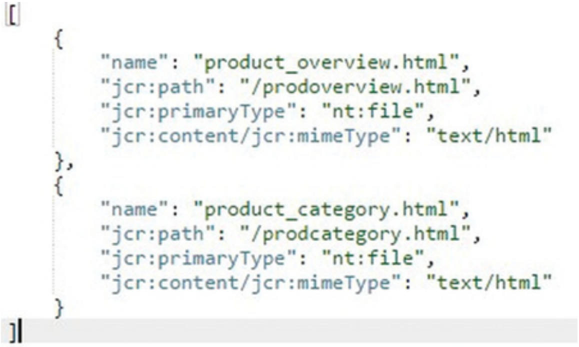
- Producing and consuming content: CMS can expose or consume content as a REST service using JSON for the clients.
- Perform content node operations: JCR and similar content repositories provide REST interfaces to perform content node operations (such as querying content node, updating content node, deleting content node, etc.)

REST is one of the widely used architecture styles for JCR repositories (such as Apache JackRabbit). Apache sling Web framework provides REST-based services for JCR content. As content is exposed as HTTP resources, we can query content, its properties, filter it, and perform other similar operations. For example, the following URL will list all JCR content nodes containing “product”:

http://server:8080/content.query.json?queryType=xpath&statement=//*[jcr: contains(.,‘product’)]

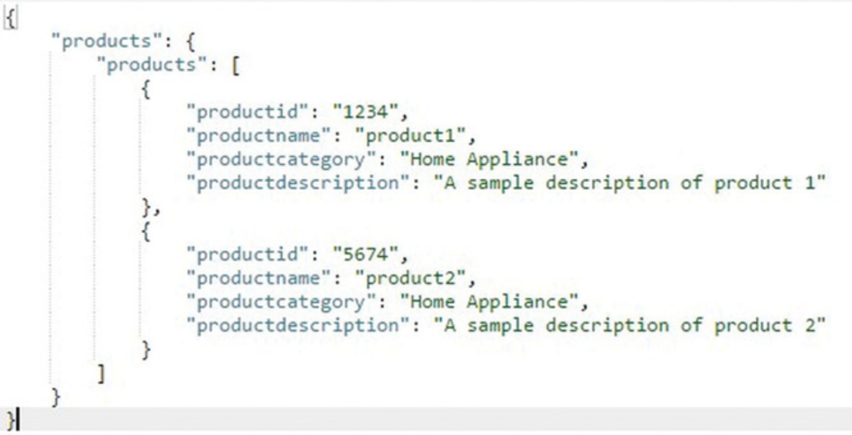
The response is rendered in JSON format and is depicted in Figure 7.16.

Another generic example of a product listing through a REST service (http://server:8080/rest-service/products) is shown in Figure 7.17.



```
[
    {
        "name": "product_overview.html",
        "jcr:path": "/prodoverview.html",
        "jcr:primaryType": "nt:file",
        "jcr:content/jcr:mimeType": "text/html"
    },
    {
        "name": "product_category.html",
        "jcr:path": "/prodcategory.html",
        "jcr:primaryType": "nt:file",
        "jcr:content/jcr:mimeType": "text/html"
    }
]
```

Figure 7.16 JSON Content Format



```
{
    "products": {
        "products": [
            {
                "productid": "1234",
                "productname": "product1",
                "productcategory": "Home Appliance",
                "productdescription": "A sample description of product 1"
            },
            {
                "productid": "5674",
                "productname": "product2",
                "productcategory": "Home Appliance",
                "productdescription": "A sample description of product 2"
            }
        ]
    }
}
```

Figure 7.17 JSON Product Listing

Applicability of SOAP and REST in content scenario

Table 7.4 provides some of the scenarios in which SOAP/REST content services are applicable.

In a nutshell, REST is a lightweight, Web-friendly architecture style that can be used while exposing content services to consumer-facing Web sites. SOAP can be used for machine-to-machine enterprise integration.

7.7 CHAPTER SUMMARY

- The main integration requirements for CMS are integration with security systems, TMS, search, publishing system, metadata system, and social and collaboration system.
- High-level security integration touch points are authentication, authorization, and single sign-on.
- CMS may be integrated with a translation management system (TMS) mainly for localization requirements.
- CMS translation workflow submits content to TMS for translation. TMS provides the job ID for the submitted content chunk. After translation, TMS invokes the CMS callback service to restart the CMS translation workflow. The workflow fetches translated content and publishes it.
- CMS can be integrated with enterprise search engine through services, APIs, search connectors, and search plugins.
- CMS can expose content and content operations through SOAP-based or REST-based services.
- CMS can be integrated with portals through portlet, CMS services, content APIs, and content widgets.

Table 7.4 SOAP and REST scenarios

Applicability	SOAP-based Content Service	REST-based Content Service
Usage Scenarios	Mainly for exposing and consuming content services through a variety of protocols (such as HTTP, TCP/IP, JMS, etc.) Content information will be mainly in XML format Most effective in distributed enterprise systems such as business process management (BPM), enterprise integrations, etc.	For exposing and consuming content services mainly over the Web and for exposing content node operations through HTTP protocol. The content information will mainly be in JSON format optimized for the Web; can also support CSV, RSS, XML Most effective in point-to-point communications
Enterprise Integrations	SOAP is more suited for B2B integration scenarios. When CMS needs to expose content service to enterprise applications or ESBs, SOAP is preferred.	REST works best for the B2C kind of scenarios. When the CMS needs to publish the content to Web channels, REST can be used, as most of the Web systems are optimized for REST and JSON.
Scalability	We need to design and test for scalability based on the real-world workload	Provides high scalability due to stateless architecture
Performance	Request and response processing is relatively costlier due to the complex nature of the XML. It is relatively heavyweight compared to REST due to overhead added by XML processing.	Comparatively lightweight, thereby providing optimal performance through caching and lower bandwidth consumption. It is possible to further optimize performance by caching REST service calls and REST reads.
Security	In addition to SSL support, SOAP also provides WS-security for supporting enterprise security features	Supports SSL and HTTP basic authentication
Transaction Handling	Supports WS-AtomicTransaction to provide ACID-compliant transactions	Supports basic transactions

- CMS can be integrated with other presentation engines through services or through published chunks.
- CMS can be integrated with MMS using APIs, services, or through JavaScript.
- Feeds can be integrated through feed plugins.
- DAM may be integrated with CMS through asset workflow.

- JCR and CMIS are popular standards that provide interoperability among compliant repositories. The key content standards are HTML/XHTML, JSON, DITA, SCORM HTML format can be used to create presentation-ready page content or content chunks XML provides a structured organization of content which can be validated against XSD/DTD DITA standard provides XML-structured content which can be categorized into topics, maps, content reference, JSON is a web friendly format mainly used for web publishing and content services SCORM specifies standard for sharing learning and training content SOAP and REST are popular web service standards used for producing and consuming web service.

Chapter 8

Digital Asset Management and Document Management

Managing digital assets such as images, media files, and other binary assets is vital to the overall success of the digital content management programs. Digital assets play a key role in increasing the effectiveness and impact factor of overall content. Digital assets play pivotal role in creating engaging, inspiring and interactive content. Many of the user testing results (such as A/B testing, multivariate testing) and Web analytics reports indicate that content with rich media assets play a pivotal role in attracting and retaining user's attention. Hence, digital assets are extensively used in digital marketing campaigns and are strategically positioned to influence user's decision. Call-to-action components and other conversion-related UI components heavily rely on digital assets to increase the chances of conversion.

In a Web content management scenario, DAM systems are mainly used to store various image resolutions that are tagged with semantic metadata. DAM systems provide such asset management features as asset creation, asset versioning, asset searching, asset uploading, asset tagging, asset distribution, asset rights management, asset reuse, asset archival, etc.

Modern consumers also expect highly relevant and engaging interactive content. Tolerance for duplicate and outdated content is very low. This is among the main reasons why enterprise digital strategy should devise methods to serve contextually relevant content with rich assets.

Managing documents is also one of the key features in content platforms. Knowledge repositories, collaboration platforms, and learning platforms heavily rely on document management features. The main concerns in document management are document acquisition (upload, creation), document sharing, document versioning, managing multiple document formats, etc. Enterprise content management (ECM) platforms provide robust document management features; most of Web content management (WCM) provides basic support for document management.

In this chapter we look at various aspects of digital asset management and document management. We begin by taking a look at the definition, objectives, trends,

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

and architecture of DAM. We also look at challenges and best practices of DAM. Following this, we look at capabilities and evolution and the road map of document management. The chapter concludes with a case study of document management solution for a banking portal.

Content architects, CMS developers, and content authors will find this chapter useful.

8.1 DIGITAL ASSET MANAGEMENT (DAM)

Digital assets include binary files, rich media files and multimedia documents (audio, video), graphics files (images, animation files, logos), brand resources (style guides), marketing material (photos, campaign creative), and the like. Here we take a look at various aspects of DAM such as business scenarios, DAM architecture, DAM challenges, and other topics.

DAM Definition

DAM is the end-to-end lifecycle management of digital assets, including asset ingestion, asset tagging, asset versioning, asset search, and asset storage and distribution. It provides a centralized repository for managing digital assets. DAM organizes the asset information and optimizes the asset functions such as viewing, indexing, searching, archiving, versioning, tracking, auditing, and the like. DAM systems provide a centralized enterprise-wide repository for managing digital assets throughout its lifecycle.

DAM Objectives and Trends

The primary objectives of any DAM system are as follows:

- DAM provides a sophisticated access mechanism to a variety of digital assets (such as images, binary files, media files, digital brochures) and provides controlled access to the assets. It should also support processing of large-sized digital assets.
- A DAM platform should enhance accessibility, agility, and value of the digital assets.
- Optimize time to market through easy-to-use integrated digital asset workflows.
- A DAM system should be able to provide custom renditions (in terms of asset resolution, size, transparency, sharpness) to suit the needs of target platform.
- DAM should be able serve and publish rich assets such as media files, graphics, and videos to create highly impactful content and improve the quality and consistency of the message.
- A DAM system should create optimal process for capturing, creating, and managing assets. The system should allow metadata tagging of assets for easier identification.

- DAM should help content systems by providing optimal assets to achieve a uniform and consistent brand experience across various channels and to create highly engaging user experience.

Some of the key trends noticed in digital asset systems are as follows:

- Usage of rich media assets to support Web 2.0 experiences. This includes supporting graphics providing immersive and engaging experience, etc.
- Easy and native integration with content management systems to provide seamless support during content authoring.
- Digital rights management (DRM) to support the integrity of the digital assets.
- Intellectual property management (IPM) to manage the intellectual property of the enterprise such as copyrights, trademark, etc.
- Federated search across multiple repositories to get the consolidated view of enterprise digital assets.
- Campaign management and digital marketing support.

Need for DAM

DAM compliments CMS and content ecosystem in realizing an effective content strategy. The main business scenarios in which DAM is useful are covered in next section. Here we look at some of key use cases that warrant DAM capability.

- If the digital assets are distributed across various applications creating asset silos and duplicate assets. DAM can be used to consolidate all digital assets into a centralized location.
- If the solution needs to efficiently secure the assets and manage access rights, then DAM is a good choice because it can offer robust digital rights management features.
- If the solution needs a large number of digital assets (images, media files, etc.) that need efficient management (such as asset workflows, asset tagging, asset lifecycle management), then DAM may be a good choice to manage the asset lifecycle.
- If asset search is an important use case for content authors, then DAM is required that can tag and search assets efficiently.
- If there are multiple asset contributors (such as creative agencies, photographers, design group) who create and edit assets, then a robust asset management system such as DAM may be needed for effective collaboration.
- If there are complex asset scenarios such as asset conversion, asset version management, batch asset processing, asset distribution, Omni-channel asset optimization, asset editing, and asset tagging, then DAM would be an appropriate for enterprises.

In light of this, it is important to understand the asset requirements during the requirements elaboration phase and to apply the aforementioned rules-of-thumb to understand the need for DAM capability. Once a decision to implement DAM has been made, we can evaluate various DAM products to find the one that fits best for our purposes. A detailed DAM evaluation template is provided in Appendix F.

DAM alternatives

Basic asset needs can be satisfied with native asset management capabilities of CMS. Alternatively we may also custom-develop a basic DAM-like simple system for basic digital asset needs. We have detailed a custom DAM approach in the "Architecting an Enterprise DAM" section.

DAM Business Scenarios

Many business areas such as marketing function, brand management function, media and entertainment industry, publishing industry, training and learning domains, public relations domain, and advertisement domain use digital assets extensively to achieve their business goals.

The following list presents some of the more prominent business usage scenarios for the digital asset management systems.

- **Campaign management:** Images and other rich media assets are extensively used in campaigns. Rich assets influence the audience's decision and generate demand for the product and services through the increased conversion rate.
- **Communication and messaging:** Assets help in communicating the information and concepts in the most effective way. Most organizations use graphics, videos, and other assets to create awareness of the new initiatives and to communicate the information.
- **Digital marketing and sales:** This is the most prominent usage scenario for the digital asset management system. Digital assets are included as part of marketing content (such as marketing collaterals, marketing documents, creative, etc.) for targeted marketing campaigns (such as advertisements, targeted mails, press releases, marketing communications, social media marketing, etc.). DAM enables the sales team to provide anytime, anywhere digital assets.
- **Brand management:** Digital assets are also heavily used in managing the brand design, brand differentiation, and competitive positioning. It can effectively convey the brand message and company reputation. Assets play a key role in the creation and management of brand assets such as logos, images, and others to create a consistent brand experience across all channels.
- **Promotion management:** Assets are also part of promotion content to convey the sales offers in the most influential manner.

There are few specialized DAM platforms based on business vertical needs:

- Brand asset management (BRM) to manage brand assets such as logos and such marketing assets

Table 8.1 DAM needs for business domains

Business Domain	DAM Needs
Advertising and promotion domain	• Rich media distribution • Campaign management • Asset and document management
Media and entertainment domain	• Ingestion and classification of digital assets • Using digital assets for promotion campaigns
Training and learning systems	• Digital rights management • Classification and categorization of digital assets • Asset search
Manufacturing	• Product asset creation • Faster development of product content • Product promotion campaigns
Publishing domain	• Efficient asset and content workflows • Asset localization
Library management system	• Digital media and digital content storage
Digital marketing	• Efficient storage and retrieval of a variety of digital assets • Various integration support options with diverse channels • Brand management support
E-commerce	• Product Image and video management • Video Search and video tagging

- Media asset management (MAM) to manage rich media formats such as videos, digital photos, and animations
- Library asset management (MCM) to manage library assets

Table 8.1 provides general DAM needs across business verticals.

Architecting an Enterprise DAM System

In this section we look at various logical components of a DAM architecture and the DAM services. A general overview of the DAM system is shown in Figure 8.1.

The main categories of components in a DAM system are:

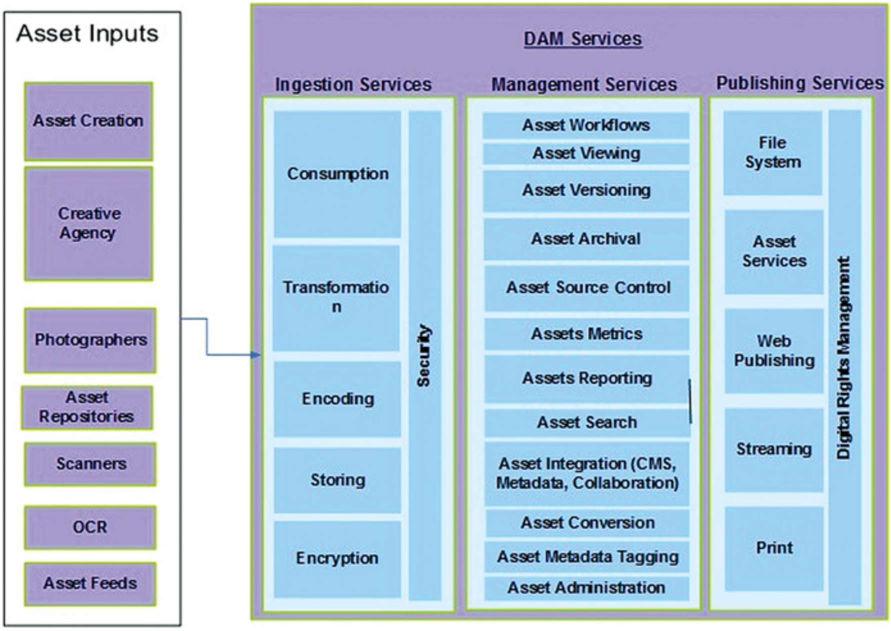


Figure 8.1 DAM Architecture and Services

Asset inputs

Assets may be provided to the system by various means. A creative agency or a design team could create new assets and upload it to DAM. Photographers may add new photos to DAM. Assets can be acquired from other asset/media repositories and file systems. Scanners and OCRs would digitize physical forms and feed digital content to DAM.

Asset ingestion components

Normally creative agencies and photographers will upload the digital assets to the DAM repository. Assets also may be acquired through scanners, video feeds, and other image repositories. DAM should be able to encode the uploaded assets and securely store it in asset repository. DAM should support acquisition of various asset formats including images, videos, audios, and similar rich media assets. It should also be able to ingest various structured and unstructured binary assets. During the ingestion process, DAM should be able to transform or encrypt the assets as needed. In some DAM systems, newly acquired assets are transformed into platform-specific binary format to enforce security policies.

Asset management components

DAM should provide efficient asset management features such as asset versioning, tagging, and viewing. It should offer asset workflows to provide various renditions of the asset and asset transformation. Asset storage and retrieval components should provide efficient storage and retrieval of the assets. DAM should provide efficient asset retrieval through caching to serve the assets.

A typical digital asset lifecycle consists of asset creation/acquisition, followed by asset editing, formatting, and transformation, and ending with asset distribution and purging. The DAM platform provides management components to cover all lifecycle stages of the digital asset.

One of the important DAM workflows is the asset-editing workflow. The editing workflow allows the asset author to edit, crop, resize, scale, and filter a digital asset. Another key asset workflow is the rendition workflow, which creates assets of various sizes and specifications. DAM developers can create custom workflows to generate various renditions of a given asset to suit their solution needs.

Assets should also be tagged (with taxonomy metadata) to facilitate relevant asset search. Digital asset metadata includes metadata values for creation/expiration date, asset type, author name, asset resolution, security rights information, etc. Similar to content, assets would also be grouped based on logical categories and stored in an appropriate folder structure.

Digital rights management (DRM) should enforce authorized access of assets. Other asset management components are asset reporting components, asset administration components, asset archival, and asset metrics tracking components. A DAM system may also provide various integration components with related systems such as CMS, metadata management, and the like.

Asset conversion is another key component of a DAM system. In many scenarios it is required to publish the asset in various formats and different renditions. To implement such scenarios, DAM should be able to convert an asset in base format into various popular digital formats.

Asset-publishing components

DAM should support various types of asset publishing such as file publishing, streaming, broadcasting, Web publishing, print media, and so on. DAM should expose various DAM functions (such as asset ingestion, asset queries) through services and APIs. CMS and other third-party systems can use these services to interact with digital assets.

Asset search service Figure 8.2 depicts a simple asset search service. Asset query services match the query tags with metadata tagged with assets to identify the best match.

Similar to content, the tags and metadata are used for searching the digital assets. In addition to regular metadata, digital assets may have additional metadata elements such as file type, image resolution, created date, and others that can be used for searching.

Asset security Assets should enforce digital rights for the assets to provide access-based controls. A digital rights management module enforces security policies

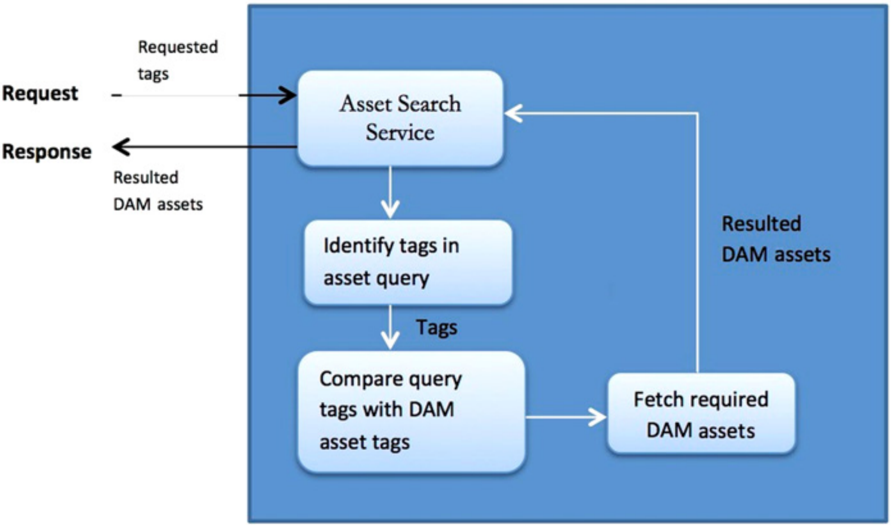


Figure 8.2 Asset Search Process

on digital assets. Secured assets should be stored in an encrypted form, which may provide role-based access to DAM functionality.

The main user roles in a DAM system are:

- **Asset creators:** These users are mainly photographers, creative agencies, and designers who provide the assets for the DAM system. Asset creators will also use scanning and other tools to acquire the images.
- **Asset editors:** These users edit the assets to suit the specifications laid out in the UX guidelines. For instance, editors create various resolutions of a single image file to cater to mobile devices. They also modify other asset attributes such as asset metadata, asset category, etc.
- **Asset consumers:** Consumers can be other systems such as CMS system that fetch the related assets while creating content.

These roles participate in the following asset lifecycle stages:

- Asset creation/acquisition, during which digital assets are created or acquired from various systems. Asset creators are actively involved in this stage.
- Asset editing: Asset editors use the asset workflow to edit the asset to suit the required specifications.
- Asset distribution, during which the asset is distributed to various channels and devices. Assets can be published as files or consumed through services.

Designing custom DAM In some cases the solution may require basic asset management capabilities such as asset storage, asset retrieval, and basic asset search. There also may not be any significant asset management requirements in the pipeline for future releases. In such cases we can either leverage the basic native asset management capabilities of the CMS or custom-build a basic DAM. We could use an efficient file storage system for storing assets (through a custom Web upload feature) and build governance processes for retrieving (through asset download feature), searching, deleting, and updating the assets from that repository.

DAM Challenges and Best Practices

The main challenges in implementing a DAM are included in the following list. Best practices that can be used to address these challenges are also mentioned.

- **Challenges due to distributed assets:** Digital assets may be distributed across various enterprise systems such as asset repository, file system, third-party systems, legacy systems, etc. Heterogeneous asset systems pose challenges during integration, management, and rights enforcement. This also gives rise to digital asset silos, which affects asset reusability. This may also lead to higher operational costs due to duplicate asset management. Without a unified and centralized DAM it would be difficult to categorize and tag assets with relevant metadata. In order to address this challenge, we need to consolidate all the digital assets into a centralized DAM platform. A consolidation exercise involves asset migration from all the asset sources to a centralized DAM platform. After asset consolidation, we need to re-tag the assets and store them in logical categories. We need to define a standard and consistent classification scheme and leverage enterprise taxonomy for consistent asset tagging.
- **Reusability challenges:** This is the side effect of distributed assets. In many scenarios the images and videos are duplicated across multiple sources, posing challenges during consolidation and classification. If more assets were distributed with potential duplicates, it would pose challenges in reusability. When assets are consolidated, duplicate assets need to be removed. Asset consolidation, categorization, and asset tagging may help in eliminating duplication and proper identification of an asset.
- **Performance challenges**: Sometimes the asset size poses performance challenges, especially while rendering on mobile devices. Assets such as videos and large binary files pose challenges in storage, distribution, and rendition and may impact overall performance. In order to address this, we need to manage assets of various resolutions for optimal performance. The asset-publishing workflow provides the optimal rendition of asset (appropriately sized) based on the target platform. Similarly, asset services should serve the optimal asset for the delivery platform.

Large-sized assets would also impact the performance of asset ingestion and processing. In order to address these issues, we need to test the DAM system on various parameters such as performance, latency, and throughput with various real-world scenarios such as batch asset processing, ingestion and processing of large-size asset, and others, and use appropriately sized hardware (using optimal memory and heap size) to handle the anticipated asset loads. Wherever possible, we should also compress the asset while sharing the asset to improve the performance. Asset caching is also another effective way to improve the performance.

- **Asset-processing challenges:** There could be excessive volume and a wide variety of assets in an organization (such as images, videos, flash files, and other binary assets). Asset volume and variety poses challenges in asset-processing activities such as asset ingestion, asset transformation, asset encoding and metadata identification, asset archiving, and the like. During ingestion of high-volume assets, we could adopt batch processing for ingesting large number of assets.

  Most DAM systems provide an efficient storage and retrieval mechanism for popular asset formats. To the maximum extent possible we need to store the assets in industry standard format and DAM compatible formats, which helps DAM manage it efficiently. If we need to store the asset in a proprietary format, DAM provides asset-processing plugins and extensions that can be used for processing of these assets.
- **Outdated and nonstandard assets:** Some of the old assets may not be relevant to the current context and would not conform to the new UX and visual guidelines. They need to be identified and transformed to meet the new standards. Wherever possible, all nonstandard assets need to be converted into DAM-compatible assets.
- **Disjointed asset business processes and workflows:** If the business processes were incompatible, it may not be possible to achieve a seamless interaction between the workflows. Asset workflows should be simplified and optimized based on asset requirements. Streamlined asset workflows should improve the collaboration between various brand teams, marketing teams, and creative agencies for smoother marketing operations and reduce the campaign lead time.
- **Asset migration challenges:** During asset consolidation and migration, incompatible asset formats and proprietary asset types may pose challenges. The digital rights, metadata tags associated with assets, should also be part of asset migration. In order to address these challenges, we should export the proprietary asset from a source system into a standard format and then import it into a target repository. Also, there should be consistent tagging across all enterprise systems using enterprise-wide taxonomy. This helps in easier migration.
- **Scalability and availability:** Many times the DAM system will not be appropriately tested for scalability, performance, and availability for the real-world

asset load. This leads to production incidents during peak load. The best way to address this problem is to perform a thorough testing of all nonfunctional scenarios (including testing for scalability, availability, performance, security, etc.) in pre-production environments. Server hardware must be appropriately sized to handle the load. We can employ the cloud hosting for DAM to provide optimal scalability and availability.

- **Omni-channel asset renditions:** In the Omni-channel-enabled world, a single-asset rendition would not be sufficient to cater to all channels and devices. A large asset size may pose performance challenges on mobile devices. In order to address this, we need to create all required renditions for an asset and use adaptive techniques to use appropriate asset rendition for a given device.
- **Asset search challenges:** Due to a wide variety of asset formats and to its distributed nature, asset search may not yield relevant results. If the assets are not organized, the search process may face performance issues. In order to improve the search performance, we need to consolidate the assets and tag them with consistent metadata. The asset workflows related to asset integration, asset storage, and asset delivery should be optimized as well.

We have so far seen various aspects of digital asset management. In the next section we look at document management in greater details.

8.2 DOCUMENT MANAGEMENT

An enterprise uses various documents for its business. The most popular documents are e-newsletters, white papers, press releases, policy documents, case studies, reports, contract documents, and the like. The document management component provides complete management of documents, including their storage, distribution, tracking, security, versioning, etc. Similar to content and assets, documents also need to be tagged with semantic metadata for efficient discovery and retrieval. Managing various enterprise documents need a robust document management system. Document management systems are mainly used for knowledge sharing and collaborative scenarios. A consolidated and standards-based document repository would help in efficient information sharing.

In this section we take a look at some of the main challenges in document management and the best practices–based design of document management platform.

Capabilities of Document Management System

We need to understand the organization's needs and business requirements to create a robust document management system. The core capabilities of a robust document management system are presented in Figure 8.3.

Document management systems should be able to manage a wide variety of documents and provide ecosystem elements for their optimal distribution. It should

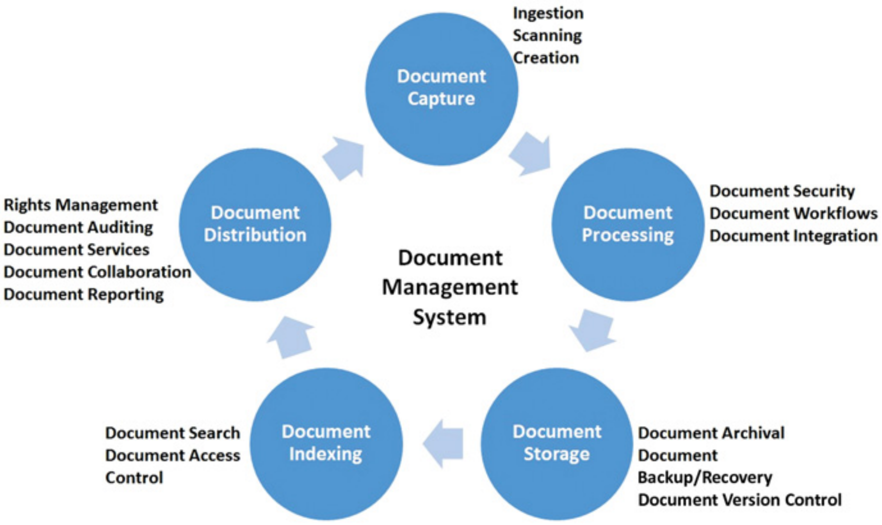


Figure 8.3 Document Management System Capabilities

be able to provide administrative features and utilities for efficient document management. The system should be able to support the following key features:

- **Document ingestion**: The system should be able to digitize (through OCR, scanning, manual upload, and other means) various formats of documents (such as policy document, process and legal documents, training documents, etc.). During the ingestion process, the system should support metadata tagging for easy retrieval in the future. The system should also help users create new documents and acquire documents from other systems.
- **Document processing**: The system should be able to encode, classify, index, search, tag, view, share, summarize, encrypt, and transform the documents into required formats. Document-processing activities usually happen through workflows.
- **Document storage** through document version control and document archival. Document storage can be used for backup and recovery.
- **Document indexing** to aid the document search process.
- **Document distribution** by publishing documents to various repositories and supporting document collaboration. The system should support document distribution in various formats and control access during the distribution process.
- **Document rights management** by allowing the role-based access to the documents.
- **Document retention** and archival as specified by regulation and business needs.

Document Management Elements and Functions

A document management system (DMS) manages creation, organization, storage, organization, tagging, and tracking of digital documents. The key functions and modules of a DMS are:

- Create and capture the documents from various sources. Enrich, format, tag, and transform the created/captured documents.
- Organize the documents so that it is easy to manage and discover the relevant documents. DMS provides various features such as document indexing, catalogs, categories, document folders, auditing, and logging to aid the organization. We can leverage workflow feature for updating, purging, and other document management activities.
- The rights management module and permissions module ensure appropriate level of user access to the documents. The document security module also may protect intellectual property.

A sample reference architecture of a document management system is given in Figure 8.4.

Components listed in the reference architecture provide the following core functions:

- User interface components such as workflow management system, search system, folder, and administration functions are used by administrators to configure the DMS for business needs. They may set up the relevant document workflows and create/configure folders for efficient document management.

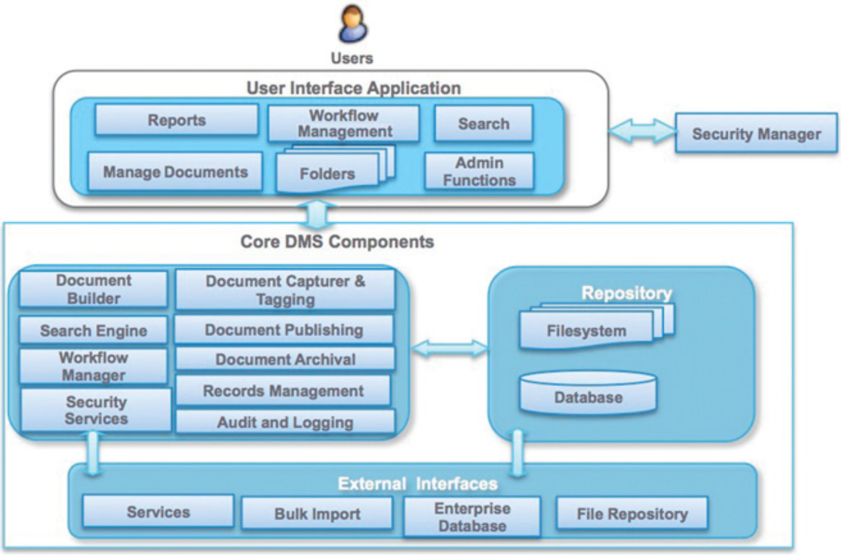


Figure 8.4 DMS Reference Architecture

- Document builder and capturer components are the core DMS modules that can be used to build or capture documents from external sources. A flexible DMS would provide capabilities to ingest documents in various formats and manage the lifecycle of the documents.
- Workflow manager helps in defining and automating business processes that involve routing documents for processing tasks, approvals, and distribution. This key component can be used for business process management (BPM).
- Document tagger defines and assigns metadata properties to documents. This helps in efficient search, storage, and retrieval of documents.
- Security services manage rights and permissions during the creation, editing, publishing, deletion, and distribution of documents. The module also provides user management feature to manage different users/roles and groups in the system.
- The document-publishing module publishes documents in various formats to a configured destination. The module also provides users with the ability to share documents with external parties in a secured manner.
- Document archival ensures that the unused documents conform to regulation requirements.
- Audit and logging track user actions in the system for future reference.
- DMS also interacts with various external interfaces such as databases, file repositories, and content services for bulk import and ingestion of documents.

Document Management Evolution and Road Map

A well-designed document management system can provide a competitive advantage for an organization. A sample road map for an enterprise document management system is shown in Figure 8.5.

The road map can be applied for asset management implementation as well. In the first stage we implement the document management solution for a specific application (such as CMS). At this stage there are no consistent standards and there will be multiple technologies and multiple processes. In the next stage we use the document management services across a wide variety of applications. We also standardize processes and technologies and tag the documents for easier discovery. In the final stage, the document management system is completely available as an enterprise service and supports advanced features such as collaborative editing and co-creation and multi-channel delivery.

Case Study: Document Management Solution for a Banking Portal

In this section we take a look at a detailed case study of using the DMS in the banking domain. The banking portal discussed in this context is used for managing policy and

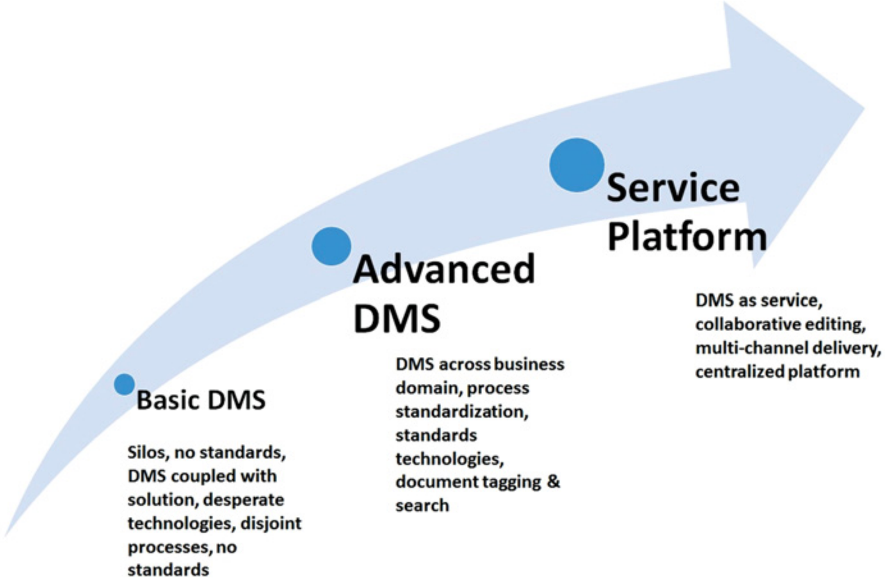

Basic DMS
Silos, no standards, DMS coupled with solution, desperate technologies, disjoint processes, no standards
Advanced DMS
DMS across business domain, process standardization, standards technologies, document tagging & search
Service Platform
DMS as service, collaborative editing, multi-channel delivery, centralized platform

Figure 8.5 Document Management System Evolution

contract documents needed for the banking activities. We explore the role of DMS in content ecosystem and various components used in DMS.

The overall solution consists of multiple applications that will work together to achieve the functionality as needed by the banking portal. Figure 8.6 depicts the logical architecture for the document management application landscape.

Let us look at various layers of the banking portal and modules in each of those layers.

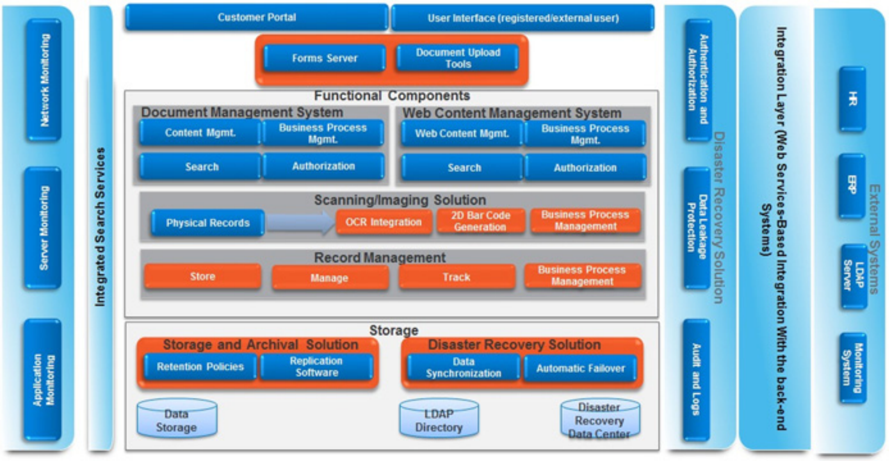


Figure 8.6 Document Management Application for a Banking Portal

Customer-facing portal and user interface

A customer-facing portal is required to provide a unified, interactive, and transactional experience to the clients accessing the site. A customer-facing portal consolidates fragmented DMS applications and provides a user-friendly interface to the client. The DMS SOA architecture exposes the DMS functionalities to third-party portals. The portal provides UI components for uploading documents to the underlying DMS.

User interface provides the desired capabilities. The two sets of users identified are Registered User and the External User. Following authentication, the existing users will employ the underlying enterprise content management system, business process manager, and compliance offerings. The external users will be provided with the capability to interact with the portal to perform self-service activities such as form submissions, request/complaints lodging, status checking, etc.

Content upload feature

The solution provides the users with an option to attach the documents to content being uploaded into the DMS. The upload process then triggers any of the business workflows that are configured, and eventually content may get stored in the records management system.

Document management system

DMS is one of the key functional components of the solution. Document management comprises all aspects of content and document management, including:

- Capturing all types of content, such as office documents, scanned images, e-mail and attachments, reports, CAD, audio, video, and business objects
- Storage and management of all types of disparate information from disconnected and stove-piped systems (meaning content can be stored, indexed, and maintained for the lifecycle of that information)
- Business process management, which can automate the seemingly endless chain of serialized, manual, repetitive, and time-consuming tasks that slow down the operation, cost money, and impede progress
- Document search feature to search for the relevant documents based on tagged metadata
- Secured access to the documents based on authorized roles and document permissions.

Access to managed content is tightly secured but easy to locate, retrieve, share, act on, and collaborate with team members, partners, suppliers, and customers.

In addition to the services listed above, DMS is configured to handle other requirements from banking domain, such as:

- Compliance requirements: The policies and workflows related to document archival, retention, and shredding must be compliant with local regulations and specific business needs.
- Auditing: All user actions and system activities related to documents are audited through audit trails.
- Security requirements: DMS supports other security requirements such as role-based access, access control lists, network level security, etc.
- Notification and reporting: DMS supports rule-based notification and activity and usage reporting. DMS keeps track of document expiration dates and provides automatic management of notifications.
- Asset categorization and management: Acts as a repository for importing, storing, and retrieving of electronic and scanned documents and other intellectual assets, classified based on content type.

Web content management system

The Web content management system allows nontechnical authors and editors to publish their content quickly and easily without having to resort to the help from technical programmers. Web publishing CMS establishes defined publishing processes/templates and specific publishing rights for various individuals. By using these facilities the solution can save time otherwise needed for training, while facilitating more people's ability to publish. It also reduces the daily stream of calls to the IT department for changes to the Web site. The Web CMS also reduces time to publish.

Scanning/imaging solution

The scanning/imaging solution enables an automatic digitization of paper documents and forms into reliable and retrievable information. The scanning solution is tightly integrated with the underlying DMS to provide comprehensive document management capabilities. The scanning/imaging solution will

- convert paper documents to reusable digital information using OCRs and scanners;
- generate 2D bar code for documents;
- streamline business process;
- apply policies and permissions automatically.

Workflow management

Whether a document is uploaded or an action is performed on any content, a workflow will trigger in the respective systems (WCMS, DMS, and RMS) to handle the business process that is defined by the application for the activity to be accomplished. The workflows will be configured in DMS and record management system (RMS) with a capability to route the request/document to the designated officers/users in the solution chain. The workflow tasks and activities can be accessed from the DMS Web-front-end application.

Record management

Record management ensures that the policies are applied to both physical and digital content. It controls the creation, maintenance, use, and disposal of records so that the right records are provided to the right person at the right time. This ensures efficiency and economy in the management of records through duplicate elimination, retention, and disposal.

Storage and archival solution

The DMS system will provide a central repository to archive a high volume of documents that can be accessed by and shared between multiple users, locations, or across the entire enterprise. Different document types and content will be archived through rights-based archival.

Disaster recovery solution

The operations of this solution are multi-site in nature. This requires a way to protect the data from disastrous occurrences at the local site. To achieve this protection, remote replication of data from the local site to the remote site is the preferred approach. Remote replication provides the benefit of restoring all the critical data, thus resuming the business even in case of site failure due to disaster.

Integration layer

The integration layer mainly consists of services based integration with back-end systems such as HR, ERP, monitoring system, LDAP system, and others.

Monitoring solution

The monitoring solution is needed to check the availability and performance of the servers and the network and ensure service continuity, as well as for the response time tracking to meet the Service Level Agreement (SLA) of the solution.

8.3 CHAPTER SUMMARY

- This chapter provides a general overview of document management system (DMS) and digital asset management (DAM).
- Digital assets include binary files, rich media files, and graphics.

- DAM is the end-to-end lifecycle management of digital assets, including asset ingestion, asset tagging, asset versioning, asset storage, and asset distribution
- Main objectives of a DAM system include providing an asset management platform for optimal asset access, providing asset flows, serving various asset renditions, tagging assets, and supporting content systems for creating a consistent brand experience.
- Digital assets are mainly used in campaign management, digital marketing, brand management, promotion management, communication, and messaging.
- A DAM platform consists of components related to asset ingestion, asset management, asset publishing, asset storage, and asset security.
- The main challenges in asset management concern asset distribution and reusability, asset performance, asset processing, outdated and nonstandard assets, asset migration, and disjoint asset workflows.
- The document management component provides complete management of documents, including their storage, distribution, tracking, security, and versioning.
- A document management system provides capabilities such as document ingestion, document processing, document rights management, and document retention and archival.
- The chapter presented a detailed case study for a document management solution for a banking portal.

Chapter 9

Content Migration

In this chapter we will build on the content management concepts we have learned so far. Content migration is an important part of the digital transformation journey, and so here we will look at various aspects of migration.

In the first section Key drivers, principles, design considerations, checklist, and challenges related to content migration are elaborated. We will then detail a migration approach and a migration example along with migration governance and evaluation of a cutover plan. The chapter concludes with a case study of JCR-based custom migration script that can be used for content migration.

Enterprise architects, CMS architects, CMS developers, and program managers may find this chapter useful.

9.1 CONTENT MIGRATION

More often than not, an enterprise will have preexisting content in various content sources such as legacy systems, old CMS platforms, Web applications, ERP systems, etc. Content from those systems cannot be used in its current form, as the new online strategy requires a different visual framework and specifies new standards. Hence, during digital transformations, when we start building a new CMS platform, we need to migrate content from those systems to a new platform. In this section we detail the concepts, best practices, steps, and other proven methodologies of content migration.

Content Migration Drivers

Enterprises normally embark on the content migration path to enable new capabilities and to enhance their offerings. Listed below are some of the main motivations for enterprises to undertake a content migration exercise:

Growth and scalability: When existing platforms fall short of handling the future business growth or when legacy content systems are not scaling up, organizations may explore alternatives for existing content systems.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

Maintenance cost: With an increase in maintenance cost of the existing systems, organizations find it difficult to implement changes to them.

Agility: Bringing in agility to business processes is a key item on a wish list for many organizations. If modern CMS can achieve it through optimal workflows and service-based models, an organization naturally may be inclined to migrate from the existing platforms.

Performance: With the increasing adoption of Web 2.0, heavily reliant on social and mobile technologies, legacy systems may pose performance challenges. Organizations may explore lightweight alternatives to enhance end-user experience.

Legacy modernization: Organization want to invest in forward-looking technologies that adapt to changing business needs. This involves modernizing the existing systems and migrating from the existing platforms to newer technologies.

Improvement in offerings: To bring in new capabilities and functionalities in an efficient manner, it may be necessary to choose the best technology landscape.

Development of centralized content system: Organizations would need to create an unified content platform consolidating content from all enterprise sources. This exercise would require content migration.

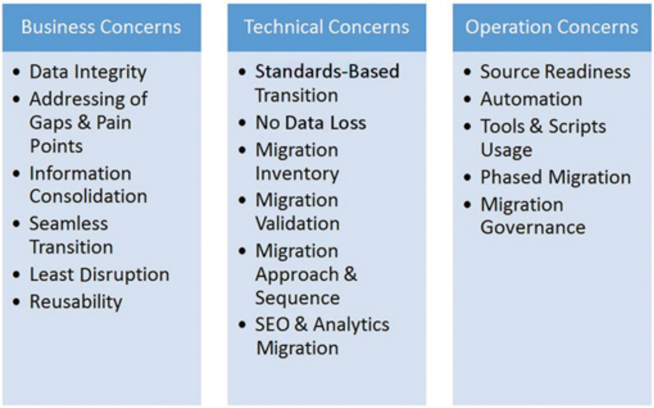
Content Migration Principles

The following list presents the key content migration principles:

- **Standards-based content transformation with minimal data loss**: During migration, sometimes we need to transform or convert content/assets across source and destination environments. In such cases it is preferred to convert content/assets to an open or standard format instead of proprietary ones. For instance, source content can be transformed into XML documents, at which point all obsolete document formats should be converted to standard format (such as PDF). This helps in efficient and flexible maintenance. Content migration should be complete with zero data loss.
- **Filling gaps and addressing pain points**: Content migration provides an excellent opportunity to address the gaps and pain points of current system. We need to compile the list of current challenges and devise a migration plan to overcome them in a target platform.
- **Source readiness:** It is always recommended to keep the source system fully ready and consistent prior to the start of migration. This includes such activities as completing any pending workflows, checking in all the documents and files, cleansing the data, ensuring the consistent metadata tagging for all content and documents, and eliminating duplicate data.
- **Migration inventory creation:** The primary step in any migration is to create an inventory of all source objects that need to be migrated. This includes content, templates, metadata, business rules, workflows, documents, assets,

etc. This inventory should be validated by all the concerned stakeholders. Additionally, we can map the source objects to target objects for each of the items on the inventory list. This helps draft the migration path and defines the process for the complete migration.

- **Phase-wise iterative and continuous migration**: It is always recommended to migrate iteratively in phases instead of doing a one-time big-bang migration. We can select the migration candidates for a given phase based on business priorities or based on the migration effort needed. For web content migration, One of the most effective best practices is to migrate the low-level site sections (such as product detail pages, Contact Us pages) in the first phase to understand the feasibility of the migration approach. Once it is successful, we can migrate the remaining site sections with an increased level of complexity.
- **SEO and analytics inclusion:** Sound SEO principles should be used to define the architecture of the target system. Content and asset discoverability, metadata strategy, and URL strategy should be considered while designing the pages, site structure, URL, and other content elements in the target environment. The migration process should ensure that existing SEO tags and scripts are completely migrated to the target platform.
- **Migration governance**: The migration governance should define the processes, checkpoints, best practices, automation opportunities, content integrity checks, content completeness checks, and verification techniques to ensure a smooth and seamless transition and complete migration with minimal downtime and with minimal errors.
- **Migration sequence**: Based on the complexity of source content and business requirements, the sequence of migration steps could vary. The migration process should clearly define the migration sequence and should comprehend data cleansing, duplicate elimination; data transformation, data validation, and all the necessary steps. While migrating the content objects from one CMS to another, the migration sequence of CMS objects is also important. We need to migrate all the dependencies (such as metadata, tags) before migrating the actual content. A typical migration sequence would be like this: taxonomy, metadata, users, user and resource permissions, database objects, digital assets, documents, workflows, templates, and finally content. By the time content is migrated, all its metadata, assets, and documents should be present in the target CMS.
- **Automation**: Wherever possible, we need to automate the migration steps through scripts and migration tools. Migration automation accelerates the migration process and enhances the quality of overall migration.
- **Information consolidation**: We need to understand if the legacy content will be restructured or consolidated during migration. This forms one of the prerequisites for legacy content migration.
- **Content reusability:** Wherever possible, we need to explore the reusability opportunities during migration. Source components that are developed in standard formats would be ideal candidates for reuse. This would reduce the rework and testing effort.



Business Concerns

- Data Integrity
- Addressing of Gaps & Pain Points
- Information Consolidation
- Seamless Transition
- Least Disruption
- Reusability

Technical Concerns

- Standards-Based Transition
- No Data Loss
- Migration Inventory
- Migration Validation
- Migration Approach & Sequence
- SEO & Analytics Migration

Operation Concerns

- Source Readiness
- Automation
- Tools & Scripts Usage
- Phased Migration
- Migration Governance

Figure 9.1 Content Migration Principles

The key principles are grouped into categories such as business concerns, technical concerns, and operation concerns, as shown in Figure 9.1.

Once we create the overall set of migration principles, we can design the migration approach and migration execution plan based on these guiding principles.

Migration Design Considerations

The key design considerations for content migration are depicted in Figure 9.2.

Migration scope: We need to finalize the scope of content that needs to be migrated. Should we migrate entire CMS content or only a subset of it? Should we migrate content as-is or should we transform it to suit the look and feel of the target site? Should we migrate content for all languages? What are the information architecture migration needs? Will the existing legacy content be consolidated or restructured or reorganized during migration? We need to find answers to all of these questions. The migration approach and methodology will be based on these items.

Migration governance: We will identify the migration processes that need to be defined as part of migration governance. We should also analyze appropriate content validation methodologies. Governance processes specify the sequence of migration steps, best-practice approaches, and post-migration activities.

Migration opportunities: We need to analyze the migration opportunities. Can we use the migration to address the existing gaps? Can we enable new capabilities such as Omni-channel enablement? Can we make the content modular and reusable during migration? We need to identify these opportunities to optimize content and end-user experience.

Migration entities: We need to compile the content inventory of all the source objects that need to be migrated. Entities include core content, content

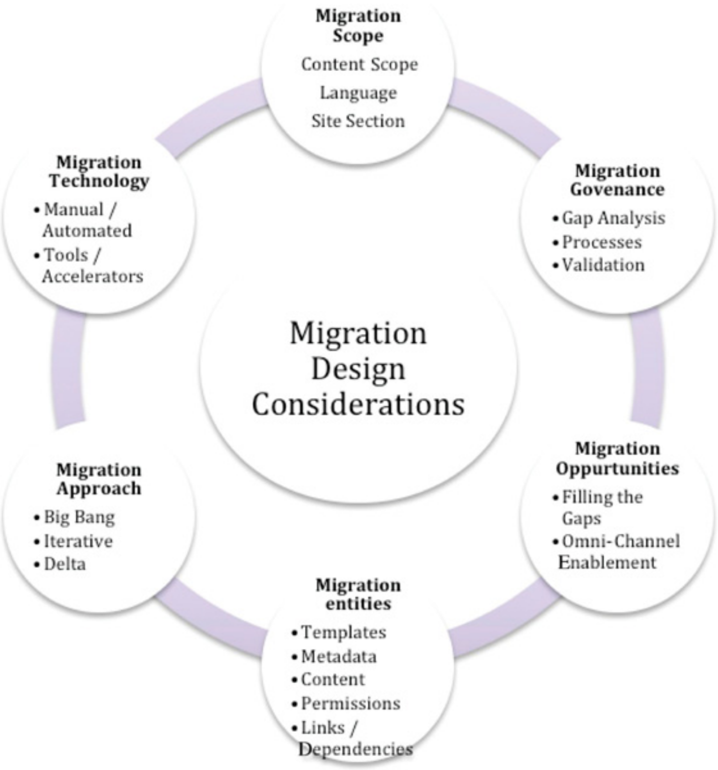


Figure 9.2 Migration Design Considerations

templates, permissions, metadata, digital assets, and other dependencies. We need to map the source objects to their equivalents in the target system.

Migration approach: We should also look at the best migration approach based on content volume and business needs. We should evaluate big-bang, delta, and iterative migration strategies to find the best fit for a given scenario.

Migration technology: We should also identify tools, accelerators, and other automation techniques to accelerate and optimize migration.

In the coming sections we discuss migration governance, migration entities and migration approach in more detail.

Migration Challenges and Best Practices

The key migration challenges and the best practices adopted are:

- **Migration scalability and automation**: The migration methodology adopted should be scalable. Sometimes the migration techniques/tools, which work for a small amount of data, may fail or perform badly during big-bang migration. This is mainly because the individual steps in the migration process are not fully tested for bulk and high-volume content. This challenge can be addressed by using a thoroughly tested migration scripts or well-established migration tools. In all cases, the migration process should be thoroughly tested.

- **Cutover plan**: This is one of the prominent challenges during migration. What is the plan for a clean cutover? Should we migrate in iterations or use the big-bang approach? If we adopt the iterative migration approach, what is the site coexistence strategy? How do we ensure data consistency during the migration period? How do we synchronize the data across new site and old site? All these points need to be analyzed on case-by-case, basis and we need to formulate a detailed plan for this. We discuss this in detail in the coming sections.
- **Completeness of migration**: Ensuring that all required content, documents, data, rules, configurations, and other elements are completely migrated is one of the critical success factors for the migration. Additionally the digital assets (such as images, videos) used in content should also be included in the migration, as well as metadata, site taxonomy, and content categories. To ensure this we must use an inventory checklist, data testing, functional testing, and scenario testing to verify if all the required content is indeed migrated.
- **Content issues and opportunities**: Over a period of time, legacy content systems may face issues such as content redundancy and may contain unnecessary content. Content flows may also be tightly tied to the legacy systems, resulting in a slower turnaround. We can use migration as an opportunity to address these challenges.
- **Dynamic content migration**: In modern applications, the page content is dynamically generated and aggregated from multiple sources. Data migration process has to ensure that all internal data sources (which are sources for dynamic content delivery) are included in migration. The libraries responsible for dynamic content delivery such as JavaScript libraries, page scripts, widgets, UI components, and adaptors should be migrated as well.
- **Data integrity challenges**: Sometimes data transformation may lead to side effects. Content or document may lose vital details during transformation, or still worse, all related content or documents may not be fully transformed. There should be sufficient test cases to catch data integrity violations, and testing should be done in each step of the migration process. A proper tracking, exception-handling, and reporting mechanism should be built into the migration solution. Migration process should be able to roll back the migration whenever it is necessary.
- **Migration approach**: We have already seen one of the side effects of a suboptimal migration approach: the migration process will not scale, or will perform badly, or will lead to incomplete migration. The best practice followed is to validate the migration approach through proof-of-concept (PoC) and validate

the feasibility, efficiency, and performance of the migration approach. In the PoC, we will also explore options to automate the migration steps.

- **Content volume:** An excessively high volume of business-critical content would pose challenges in validation. A validated migration approach with a good amount of automation to support batch and iterative migration would be required to address this challenge.
- **Management of various environments**: During the process of migration, it may be challenging to manage multiple environments and keep them all in sync. A well-designed migration plan and cutover plan may be needed to manage all environments.
- **Usage of proprietary and third-party technologies in source systems**: If there are no equivalent components in a target system for proprietary source system components, then we may end up developing custom components based on open standards in a target system. It is always recommended to migrate to standard content formats so that we can effectively leverage tools and open-source framework for migration.
- **Special content challenges**: In some scenarios existing content may be highly customized, which may pose challenges in migration. In such cases automated content transformation/migration tools cannot be used to automate the migration fully. Some of them are:

  1. Images used for special symbols and characters that may need manual transformation
  2. JavaScript-based links within content

  In such cases content cleanup and reconciliation needs to be done before using the content migration tool. The tool also needs to be customized to suite the migration needs
- **Content and asset corruptness**: During the course of migration, content (especially multi-byte content) or binary asset may get corrupted. In order to avoid this, we need to do a pilot migration involving all unique content types and assets to test whether the migration tools and processes correctly handle the required content types and assets.
- **Naming conventions:** Content and file naming conventions may not be compatible across source and target systems. While the source CMS allows the usage of special characters and separators as part of a file name, the same may not be supported in the target environment. We need to identify such special scenarios and change the naming patterns accordingly during migration.
- **Data model migration**: Each CMS has its own data model to store the content information. They may also use an internal database to store the content (along with its metadata) information. When we are migrating from one CMS to another, the data models may not be compatible. The best way to address this problem is to bring both the source and target data models into a common, uniform, and compatible form. Most of the CMS can export the data model into a standard format such as XML. We can apply the transformation rules to

Table 9.1 Migration best practices by content type

Content Type	Example	Migration Best Practices
Text Content	HTML content	Use open standards such as XML, DITA, or JSON to migrate text content Use batch migration Use tools to automate the content transformation and migration
Digital Assets	Images, video files, binary documents, audio files	Always store the assets in an external asset repository, such as a DAM system, instead of a proprietary CMS repository Target CMS can connect to the centralized DAM for asset needs
Documents	PDF, MS Office documents	Preferably use a centralized document management system or document repository and integrate CMS with that system
Metadata	Tags, keywords, terms	It is recommended to use centralized enterprise-wide metadata management system and integrate CMS with it.

that XML to make it understandable (and importable) into the target CMS's data model.

- **Dealing with unnecessary content:** In most of the cases, source systems tend to have a lot of content that should not be included in migration. Duplicate, outdated, and incorrect content in a source system needs to be identified and removed from the migration scope. This will eliminate the need to migrate unnecessary content and thus speed up the migration process. The source objects not considered for migration should be properly documented in the reconciliation report, along with reasons for leaving them out.

Table 9.1 lists migration best practices based by content type.

Migration Checklist

For complex migration scenarios, we will also maintain a checklist that is reviewed and signed off by all stakeholders. A sample content migration checklist is given in Table 9.2.

Migration Approach

In this section we take an in-depth look at the migration approach. We look at the migration sequence that uses previously discussed best practices.

Table 9.2 Migration checklist

Description	**Y/N /NA**
Contact List	
The migration team should contain a contact list of all concerned stakeholders, SMEs, and content architects	
Content Migration Requirements	
All migration requirements are documented and signed off on by stakeholders	
The migration phases and priorities are finalized and signed off on by stakeholders	
Nonfunctional Requirements	
The content volume and content availability during the migration is documented and signed off on	
The expected performance of the migration is documented and signed off on	
All the nonfunctional service level agreements (SLA) for the migration are documented and signed off on	
The content migration security requirements are documented	
Migration Impact	
The affected internal and external applications and services are notified	
CMS users and end-users are notified about the migration impact	
Content migration impact is documented	
Content Migration Approach	
Content migration approach is fully documented in the approach document	
Risk, mitigation plan, and contingency plan are documented in the approach document	
Migration phases are identified, prioritized, and signed off on in the approach document	
The plan for coexistence of old content platform and new CMS is fully documented along with related content synchronization procedure	
All content migration optimization procedures are identified, including migration automation, automatic conversion, batch migration, etc.	
All the necessary migration tools, frameworks, and scripts are identified and procured or developed	
Migration approach and feasibility are validated using migration proof-of-concept (PoC)	
All possible reusable content, tools, and systems are identified	

(*continued*)

Table 9.2 (*Continued*)

Description	**Y/N /NA**
Content Migration Testing	
Migration testing scripts are developed	
Exhaustive testing steps to verify the completeness and accuracy of migration are documented in a migration validation document	
Content validation test cases are developed	
Content security test cases are developed	
Release Management	
The release plan (EOL plan, coexistence plan, cutover plan, site synchronization plan) is identified and detailed	
Content from the source systems is fully backed up to support the contingency plan	

The migration process is customized for each of the environments and business scenarios. Main phases of migration approach are given in Figure 9.3.

Discovery phase During the discovery phase, we analyze the existing system and create an inventory of all content that needs to be migrated. We then create a detailed migration strategy and a migration plan.

Development phase Before executing the actual migration, we will execute a proof-of-concept (PoC), followed by a pilot migration in the development phase. Pilot content migration can be done for medium-complexity content stored in a source repository. Pilot migration will have the following stages:

- Batch Migration – The pilot migration batch should be small in terms of content volume but addressing different complexities identified during the requirement

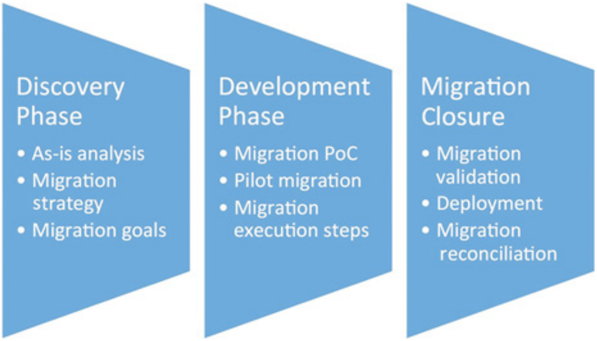


Figure 9.3 Content Migration Approach

analysis phase – for example, document linking, indexing fields with special characters, multiple versions, etc. This helps in understanding the issues and complexities that may occur in the production environment before production migration.

- Verification – This stage involves verification of whether all content in the batch has been successfully migrated from source to target repository.
- Delta Migration – This stage involves migration of newly created content and content already migrated but updated (change in metadata or new version added, etc.) during time of batch migration. It also involves migration of content that failed during the batch migration.

We fine-tune the mapping information, migration utility, and migration scripts based on the experience in pilot migration.

Migration closure phase In the migration closure phase, we migrate production content (content, digital assets, metadata, components, etc.) in batches. We then test the migration for completeness. Migration reconciliation is one of the main closure activities that provide the audit of migration activities.

Detailed migration steps in this phase are discussed in the next section.

Detailed migration steps As a part of assessment and migration execution phases as discussed in Figure 9.3, here we elaborate on the migration steps for a given scenario. The detailed steps in those phases are shown in Figure 9.4. We validate these steps during PoC and pilot migration phases.

Migration inventory creation and mapping of migration candidates The first step is to compile all the content sources needed and define the scope of migration. During this step we analyze and create an inventory of candidate objects for migration. The inventory list should be exhaustive to include all static content, dynamic content, digital assets, associated components, dependencies, etc. During the process we need to look at existing content and categorize it into categories such as "migrate," "discard," "outdated content," "duplicate content," "revise and migrate." Include only content that belongs to the "migrate" or "revise and migrate" category. A sample inventory list consists of the following elements:

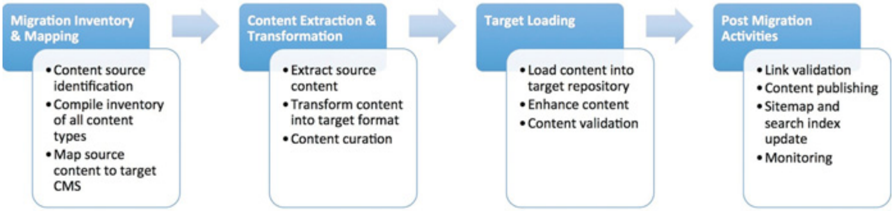

Migration Inventory & Mapping
• Content source identification
• Compile inventory of all content types
• Map source content to target CMS
Content Extraction & Transformation
• Extract source content
• Transform content into target format
• Content curation
Target Loading
• Load content into target repository
• Enhance content
• Content validation
Post Migration Activities
• Link validation
• Content publishing
• Sitemap and search index update
• Monitoring

Figure 9.4 Migration Steps

- HTML content
- Metadata
- Documents
- Templates
- Workflows
- Documents
- Pages
- Site structure
- Digital assets such as images, videos, etc.
- Localized content
- Versions
- Existing user roles and responsibilities
- Audit history
- Asset and content versions
- User permissions
- User group mapping
- Configurations
- Any dependencies between master content with its assets and localized versions.

During this step we also map the content type in a source system and a target system. For instance, workflow steps in the source system will translate to workflow steps of different nature in the target system; similarly the metadata in the source systems may be mapped to new metadata structure in the target environment.

A sample mapping of various content types between Drupal and JCR systems is shown in Table 9.3. The table also provides the migration plan for each of the content types.

Table 9.4 provides the support for various CMS features in popular CMS platforms. This helps us during migration for mapping content objects between source CMS and target CMS.

Note: The feature mapping in Table 9.4 is for educational purposes only. Plugins, features, and support modules are constantly evolving, and hence this table would not fully capture all the capabilities of the listed CMS.

Content extraction and transformation The next step is to extract content and all its dependencies (metadata, assets, versions, annotation, documents) from source CMS and apply the transformation rules so that content is converted into desired format on target environment. We could use the source-target mapping document we had prepared in the previous step.

This extract-transform is done on an iterative basis and is tested in a staging environment.

Table 9.3 Content type mapping across Drupal and JCR based content repository

	Source CMS (Drupal)	JCR Content Repository (Apache JackRabbit)	Migration Plan
Text content (blog, HTML, text, articles)	Content nodes stored in a database	Nodes and properties (named property indicates the data type)	Script- or tool-based migration
Non-text content (images, videos, assets)	Stored in a Drupal repository as binary files	Nodes and properties (named property indicates the binary content type)	Extract the assets from the repository and batch-load them into a target system
User profile attributes	Permission model using users table	JCR permission model and resource-based ACLs	Leverage centralized user management and provisioning system
Metadata	Stored in a database	Node properties (nt:*)	Leverage centralized metadata management system and re-tag content
Site and user experience design	Theme component	Not applicable, as JCR is a content repository	Preferably redesign target user experience and use target CMS specific features of it
Workflow	Workflow module of Drupal	Not applicable, as JCR is a content repository	Needs to custom-build the workflow

Content restructuring/enhancement, curation, and content cleansing happen at this step.

As this is the core step in migration that processes the bulk of content, there is a huge scope for automating activities of this step. We look at the automation aspect in the next section.

Content loading into target repository Once the transformation is over, the transformed content is loaded into new environment through an automated process. Wherever required, we will also enhance and re-validate the content. Once all the steps are completed, the content is published.

Content Migration Examples

Let us look at the two examples of a CMS-to-CMS content migration.

Table 9.4 CMS features mapping

	Drupal	Joomla	WordPress
Content creation	Content types, fields, nodes Support for rich text editor (RTE) in content creation	Content manager, content types (articles, banners, contacts, newsfeeds, weblinks)	Custom post types, custom fields, custom taxonomies
Page layout	Views module	Views	Page templates and themes
Permission model and access control	Built-in role and permission model	User, groups, and permission management feature	Built-in roles and access manager
Social features (blogs, wiki, forums, communities, polls)	Forum module Polls	Discussions extension for forum	Built-in robust blogging platform
Forms	Webforms, form builder	Proforms	Gravity forms
Multilingual components	Built-in support for multiple languages	JA multilingual component and extensions	Built-in support for multiple languages
Rich media management	Media module	Extensions such as K2	Built-in media library
Workflow	Built-in revisioning support	Built-in versioning support	Plugins such as edit flow

Note: The migration examples given are for education purposes only. Actual steps may vary based on the complexity of migration.

A sample list of activities for migration to Drupal CMS is depicted in Figure 9.5.

As a first step, we compile the migration candidates from a source system and extract the identified candidates. In the transformation step we create the content object mappings for Drupal CMS and perform data transformation. During the load

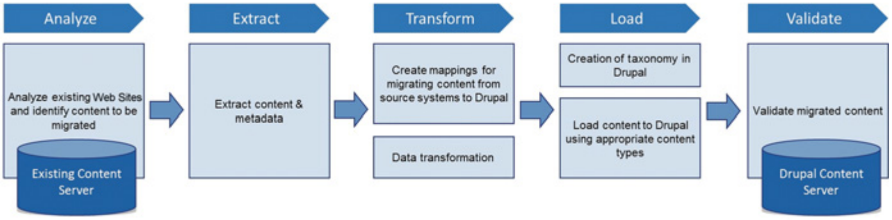

Analyze
Analyze existing Web Sites and identify content to be migrated
Existing Content Server
Extract
Extract content & metadata
Transform
Create mappings for migrating content from source systems to Drupal
Data transformation
Load
Creation of taxonomy in Drupal
Load content to Drupal using appropriate content types
Validate
Validate migrated content
Drupal Content Server

Figure 9.5 Migration to Drupal CMS

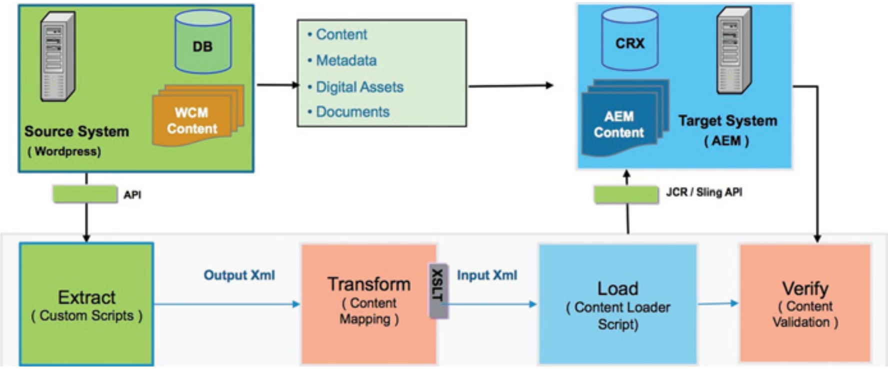


Figure 9.6 Migration to Adobe AEM

phase, we create taxonomy in Drupal CMS and load content into Drupal using appropriate content types.

Another general migration sequence from WordPress to Adobe AEM is depicted in Figure 9.6.

In this scenario, the extraction script uses custom API to get all WordPress content (including metadata, digital assets, and documents), which then is mapped to target content types in AEM. During the load phase, a custom load script loads the XML-based data into the AEM JCR repository. We will take a look at Java-based code for JCR node creation in the migration case study.

Note: Migration steps depicted in Figures 9.5 and 9.6 are general steps. Based on the content types, formats, and content model, the steps and complexity of migration may vary.

Post-migration activities The key activities post-migration are:

- Content validation: The validation team needs to ensure that all the content items are migrated (and transformed) as expected. Completeness and integrity of content are validated.
- Migration reconciliation report: Once the migration is complete, we need to create a report of all the source objects that were migrated, objects that skipped the migration, objects that got transformed, and objects that could not be migrated. The report is important for business stakeholders to understand the migration effectiveness. We also need to identify the reasons for skipped migrations and migration errors and develop a strategy to re-migrate them.
- Link validation: We also need to validate the links within content to ensure they are not broken.
- URL redirects: We can decommission the old Web pages or configure the URL redirects (sometimes referred to as 301 redirects) to new pages.

- Search indexes: We need to clean up the old search index and configure the search crawlers to crawl and index the new content.
- Publishing: Once the migrated content is validated, it can be published to the live site.
- Synchronization: In some scenarios both old and new sites will coexist for some time until the old site is completely migrated. In such cases we need to synchronize the data between them to keep them in sync.
- Error page: Create a friendly error page (also known as 404 page) for invalid resource URLs. Provide suggestions and a search feature for users to explore and identify the correct resources on the new site.
- Sitemap update: Update the sitemap with new page URLs and update this information in other required pages (such as seed list page, site robots page, etc.).
- Security validation: Ensure the necessary security constraints, permission model, and role-based access are working as expected on the new site as well.
- Validation of SEO and accessibility: On the new site, we can check if the SEO, Web analytics, and accessibility work as expected. We can check the Web analytics reports to see if the new content and assets are tracked as expected.
- Performance benchmark against the baseline: Verify the key metrics (such as page load times) against the baseline metrics obtained prior to migration to quantify any improvements.
- Monitoring based on metrics: We need to continuously monitor the key metrics using Web analytics and site-monitoring tools. This ensures optimal response in case of production incidents. We can also measure the impact of new content on user experience through metrics such as site traffic, time on site, exit rate, conversion rate, etc. This helps in quantifying the improvements caused by new content.

Migration Governance

Governance spans the entire lifecycle of content. Even during content migration, content governance defines the main processes to ensure that content migration follows the best practices:

- Content governance provides the content gap analysis of existing content and the desired end state. Gap analysis is done by benchmarking existing content with industry standards, competitive content analysis, and based on end-user feedback/expectations.
- Gap analysis phase of content governance would also provide crucial inputs to streamline the workflows and processes in the target environment.

- Governance also defines a tracking mechanism (different from migration testing) to ensure that all source content is mapped to corresponding components of the target CMS. This acts as a pre-migration checkpoint to ensure the completeness of source elements, their inclusion in the migration process, and mapping to target elements.
- Evaluation and selection of migration tools is also one of the key activities defined by content governance. We consider various factors such as suitability of the tool to the current migration context, degree of automation that can be achieved, scripting capabilities, support for configuration/ manual migration, and other capabilities needed to arrive at the right set of tools.
- Content governance also decides the migration phases. It specifies the migration approach (iterative or one-time), prioritizes the content for each migration phase, and defines the migration scope for each phase.
- Content governance also defines the post-migration review process. The review process includes migration-testing steps, verbiage checks, broken links checks, document count check, redirection checks, SEO checks, search results checks, etc.

Migration Automation

The opportunity for migration automation is mainly in the read-extract-transform-load-validate stages as given in Figure 9.7.

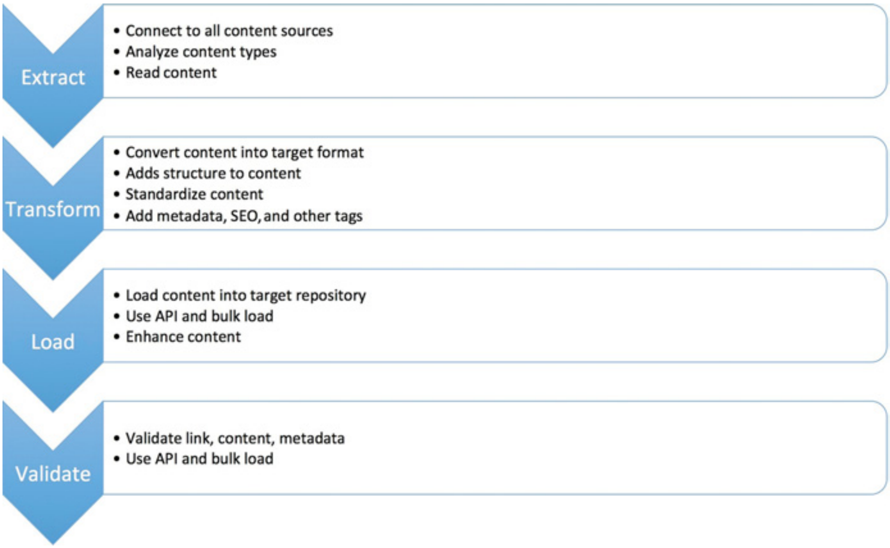

Extract
• Connect to all content sources
• Analyze content types
• Read content
Transform
• Convert content into target format
• Adds structure to content
• Standardize content
• Add metadata, SEO, and other tags
Load
• Load content into target repository
• Use API and bulk load
• Enhance content
Validate
• Validate link, content, metadata
• Use API and bulk load

Figure 9.7 Migration Automation Stages

The key steps of automated migration are:

- **Extraction:** The migration script should connect to all enterprise data sources to get the source content in the inventory list. The data sources include content systems, file servers, metadata systems, digital asset systems, document repositories, services, unstructured content storage systems (such as chat, blog, e-mail), etc. During extraction, the scripts should be able to extract and categorize content. For instance, the extraction script should recognize the content type, metadata tags, documents, binary assets, XML content, JavaScript libraries, collaborative content (for blogs, chat, e-mail), etc. This structuring and categorization of content acts as precursor for future transformations. At the end of this step, all enterprise content is extracted and categorized into predefined content categories.
- **Transformation:** This step bridges the gap between a source system and a target system by transforming extracted content into the required format. Often target systems specify a predefined content structure (such as XML with defined schema, JSON, etc.) and we transform extracted content into structure compatible with the target system. We will configure the transformation rules and mapping rules in a configuration file in this step. Transformation engine uses the rules file to perform transformation. The main activities in this step are:
  - Converting source content into a standard format (normally XML) compatible with target content system
  - Assigning content to new content categories present in the target content system
  - Assigning the new metadata (supported by target content system) to content
  - Curate content through content cleansing. It involves activities such as fixing broken links, fixing misspellings, replacing outdated content, fixing download issues, fixing content gaps, removing duplicate content, converting content into structured format, and the like.
  - Any errors encountered during transformation will be logged so that we can retry the transformation after correcting the errors.
  - Changes required for target CMS such as date format change, cleanup of special characters, file format changes, etc.
  - At the end of this step, the source content is fully transformed into a format compatible with the target content system.
- **Load:** The transformed content is loaded into the target system in this step. Content loading can happen through:
  - Content loading APIs (such as JCR API) supported by the target content system
  - Leveraging the Web interface of the target content system
  - Service calls to load the target content system
  - Using third-party content loaders
  - Direct load into the target content repository

  In all these cases, a batch load is employed for optimization purposes.

- **Validation:** Once the target system is loaded with new content, we can perform the required validations to ensure completeness and integrity of content migration.

For extraction, transformation, and load we can use commercial ETL tools or we can develop custom scripts using the vendor-supported content APIs. The entire migration solution should be highly configurable and scalable so that the same framework (fully or partially) can be reused in other migrations across the enterprise. The individual components like extractor, transformer, loader, validator, and logger should be configurable. At each stage of the process, we need to log the activities so that it is easy to track and report the migration activity.

We could create a migration framework using API-based custom scripts for automating migration from source repository to target repository. A sample API-based migration automation approach is depicted in Figure 9.8.

A migration framework essentially consists of an extractor framework, staging-area transformation, and a loading framework. Extractor framework analyses content (automatically identifies content types, files, navigation elements) and maps identified content types to target content type. It would then extract content from source CMS using supported APIs. Extracted content is transformed into XML documents. XML documents are enhanced in intermediate staging area as part of the transformation step. The transformation step includes addition of metadata, content standardization, duplicate removal, and content cleansing. Cleansed content is loaded into target CMS using the loading framework. Content is loaded using content APIs supported by target CMS. After content loading, it is validated to ensure the integrity of migration.

Cutover Plan

One of the important considerations post-content migration is the cutover plan, which cleanly separates the old system with the new CMS. The following list presents the popular cutover strategies:

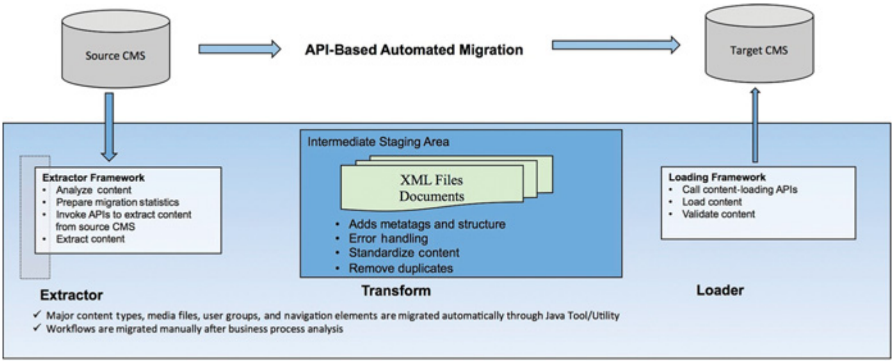

Source CMS
API-Based Automated Migration
Target CMS
Intermediate Staging Area
XML Files
Documents
• Adds metatags and structure
• Error handling
• Standardize content
• Remove duplicates
Extractor Framework
• Analyze content
• Prepare migration statistics
• Invoke APIs to extract content from source CMS
• Extract content
Loading Framework
• Call content-loading APIs
• Load content
• Validate content
Extractor
Transform
Loader
✓ Major content types, media files, user groups, and navigation elements are migrated automatically through Java Tool/Utility
✓ Workflows are migrated manually after business process analysis

Figure 9.8 API-Based Migration Framework

- **Big-bang migration and big-bang cutover**: In this case the entire content and data from the old system is migrated completely and the system itself is allowed to sunset. This involves one-time complete migration from the source system to the target system. This requires high amount of testing to ensure that content is fully migrated and hence involves high risk. This approach can be employed when we are doing simple migrations wherein the source system has minimal content and when the migration process does not need detailed validation.
- **Iterative migration and gradual cutover:** In this case, content migration and testing happens in various iterations. Each iteration scope is based on business priority and migration complexity. For instance, in the first stage the less complex content (such as static content) will be migrated and all the static pages will redirect to the new site. In subsequent iterations we target more complex content, which requires transformation. Another example could be that in the first iteration we target content belonging to one geography or language and in subsequent iterations we target other geos and languages. Similarly, site sections can be migrated one by one through iterations. We select a logical group for each iteration in the same fashion. Iterative migration is adopted during complex migrations, which involve high volume of business-critical content, and iterative testing provides high level of confidence in effective migration. Iterative testing reduces the risk, as we can catch the errors in initial iterations and make subsequent iterations more robust.
- **Delta migration**: During iterative migrations we also perform delta migration to enhance performance. Delta migration includes migrating only the "differences" since previous migration. Delta migrations are less costly and hence can be done frequently. Delta migration is one of the most effective techniques in site synchronization because it causes minimal overhead on source and target systems.
- **Business criticality based migration**: In this scenario business-critical pages and functionalities are targeted for initial iterations. For instance, all content and associated assets on home page and product pages will be considered for the first iteration, followed by product details pages. The iteration candidates are based purely on business priority. This is adopted when business wants to roll out the new platform quickly to the market to gain a competitive advantage.

Comparison of various migration strategies is presented in Table 9.5.

Migration Case Study: JCR-Based Custom Migration Script

Due to complex nature of migration, we will adopt a combination of manual, tool-based, and script-based approaches for end-to-end migration. Content inventory creation and mapping is a manual step, while extract-transformation-load can be achieved

Table 9.5 Migration approach comparison

Method	Pros	Cons
Big-Bang Method – Involves one-time bulk migration. Validation happens post migration.	• Content and dependent objects can be migrated and made available all at once • Best suited for small-volume, less complex, and mainly static content migration • No need to design a site coexistence strategy	• Longer migration cycles, which could impact users • Relatively higher risk, as validation happens post migration. • Is not suited for high-volume and complex content migration
Iterative Mode Method – Migration happens in logical iterations. Each iteration includes content belonging to a particular site section, business department.	• Faster migration cycle • Quicker discovery of issues due to iterative testing • Reduced risk	• The approach might cause additional overhead for simple Web sites • We need to design a site coexistence strategy, which can pose additional synchronization challenges.
Delta Method – The differences from previous migration are migrated. This can be carried out frequently.	• Quicker migration cycles • Faster testing cycles	• Can be used only for subsequent migration phases • We need to design a robust mechanism to properly identify changes in content and its dependencies

through tools or through scripts. Migration validation can also be automated through test scripts.

In this case study we discuss a script-based migration to a JCR system. We develop script to perform a simple migration.

Note: The case study uses Java for migrating to a JCR system. Migration scripts are normally developed using the content APIs and services exposed by the target CMS.

JCR migration script Migration scripts provide greater control over migration steps and can be effectively used for small-scale migrations. What follows is an example code for migration across JCR repositories. Most of the commercial content repositories are JCR compliant, and hence if we devise the migration script based on this JCR standard, then it will be easy to reuse. In a JCR content repository, content is stored as JCR nodes along with its properties. We use JCROM, an open-source annotation-based framework for developing the migration script. In the example that follows, we migrate a "Person" content node from a source JCR repository to a target repository. The main steps in the migration are as follows:

- Create a Java POJO class to model the JCR nodes. The properties should be annotated.
- Read the node details from the source JCR repository into a Java object.
- Write the details from the Java Object into the target JCR repository.

Step 1: Create an annotated java POJO class to model the JCR person node

```
import org.jcrom.annotations.JcrChildNode;
import org.jcrom.annotations.JcrName;
import org.jcrom.annotations.JcrPath;
import org.jcrom.annotations.JcrProperty;


public class PersonDetails {


        @JcrName private String id;
        @JcrProperty private String name;
        @JcrProperty private String emailId;
        public String getId() {
                return id;
        }
        public void setId(String id) {
                this.id = id;
        }
        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name = name;
        }
        public String getEmailId() {
                return emailId;
        }
        public void setEmailId(String emailId) {
                this.emailId = emailId;
        }
}
```

Step 2: Read the "person" node from a source JCR repository into the Java POJO class

```
public PersonDetails readFromSource(){


                Session session = null;
                Jcrom jcrom = new Jcrom();
                //Mapping java class
                jcrom.map(PersonDetails.class);
                try {

                       Repository repository = JcrUtils.
                       getRepository("http://sourceserver/server");
                       //Getting Administration session
                       session = repository.login(new
                       SimpleCredentials("adminname", "adminpwd".
                       toCharArray()));
                       //Read Employee node
                       Node person = session.getRootNode().
                       getNode("Person").getNode("12345");
                       PersonDetails personDetails = jcrom.fromNode
                       (PersonDetails.class, person);
                       return personDetails;
                } catch(Exception e){
                       Logger.getLogger().error("Exception occurred
                       in reading nodes "+e.getMessage());
                }finally {
                       if(session != null){
                                  session.logout();
                       }
                }
 }
```

Step 3: Create the "person" node read from the source JCR repository into the target JCR repository

```
import javax.jcr.Node;
import javax.jcr.Repository;
import javax.jcr.Session;
import javax.jcr.SimpleCredentials;
import org.apache.jackrabbit.commons.JcrUtils;
import org.jcrom.Jcrom;
public void writeToTargetJCR(){
                 Session session = null;
                 Jcrom jcrom = new Jcrom();
                 jcrom.map(PersonDetails.class);
                 //Populate Basic details
                 PersonDetails personDetails =
                 readFromSource();
                 try{
                        //Configuration :: Repository Details
                        Repository repository = JcrUtils.
                        getRepository("http://targetserver/server");
                        //Getting Administration session
```

```
                       session = repository.login(new
                       SimpleCredentials("adminuser", "adminpwd".
                       toCharArray()));
                       //Get the parent node
                       Node parentNode = session.getRootNode().
                       addNode("Person");
                       //Add basic node
                       jcrom.addNode(parentNode, personDetails);
                       //Don't forget to save session, otherwise
                       changes won't reflect
                       session.save();
                }
                catch(Exception e)
                       Logger.getLogger().error("Exception occurred
                       in writing nodes "+e.getMessage());
                }
                finally{
                        if(session != null){
                         session.logout();
                        }
                }
}
```

9.2 CHAPTER SUMMARY

- The main content migration design principles are standards-based content transformation, filling gaps and addressing pain points, source readiness, automation, migration inventory creation, phase-wise iterative and continuous migration, SEO and analytics inclusion, migration governance, and information consolidation.
- Key migration challenges are migration scalability and automation, cutover plan, completeness of migration, content issues and opportunities, dynamic content migration, data integrity challenges, migration approach, content volume, management of various environments, usage of proprietary and third-party technologies in source systems, special content challenges, content and asset corruptness, naming conventions, and data model migration.
- Key migration steps are inventory creation and mapping of migration candidates, content extraction, and transformation and content loading
- During inventory creation and mapping of the migration candidate step, we audit and compile all the migration candidates and map it to target CMS equivalents. During content extraction and transformation steps we use tools and scripts to extract content from source system and transform it into format compatible for target CMS, and finally during the content-loading step we load the transformed content into target CMS.

- Migration governance involves content gap analysis for migration candidates, content tracking, and migration tool evaluation.
- Extraction, transformation, load, and validation are ideal candidates for automation.
- Main cutover strategies are big-bang migration, iterative migration, delta migration, and business critical migration.
- Big-bang migration migrates entire content at once; in iterative migration, we migrate content in iterations, in delta migration we migrate only the differences, and in business critical migration we will migrate content based on its business criticality.
- The chapter provides a JCR-based migration script case study. In this case study we model a JCR node using a Java POJO class and use JCR APIs to read from a source repository and then write the details into a target repository.

Chapter 10

Content Governance: Validation, Analytics, KPIs, SEO, and Evaluation

In this chapter we mainly look at various aspects of content governance. Content governance involves various activities and we focus on some of the key topics in this chapter. Key governance aspects discussed in this chapter are content validation, which covers various content validation methods; content analytics and KPIs, which help in tracking and monitoring content effectiveness; and content SEO to enhance content visibility and content evaluation. Content governance also involves other activities such as definition of role-responsibility matrix, definition of content processes and optimization of content operations. We explore these in other chapters. We need a comprehensive evaluation framework to shortlist and finalize a CMS product for a specific solution domain. This chapter offers a look at a proven CMS evaluation framework. Appendix C provides the CMS evaluation template that compliments the evaluation framework detailed in this chapter.

Validating content from various dimensions is an important aspect of ensuring content quality. Content and the CMS should be subjected to various kinds of validation to ensure that the content platform scales and performs well. We need to simulate all possible real-world scenarios to conduct robust testing.

We have seen the importance of content analytics while discussing content strategy and content architecture. Content analytics plays multidimensional roles such as tracking content metrics, aiding in personalization, gathering user insights, and gauging content effectiveness. Here we discuss content analytics and important content KPIs.

SEO brings visibility to the content and greatly helps marketing efforts. We will also discuss content SEO and SEO best practices.

We begin with a look at content validation topics along with content analytics, KPIs, and content SEO. We discuss various testing types along with their objectives

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

and approaches. We also look at a content checklist that may be used as gating criteria for comprehensive content validation. We then look at scenarios for content analytics and look at the impact of content analytics on content strategy. We will explore a content analytics case study for an e-commerce site along its various stages. Various content KPIs that can be used as tracking metrics are provided in the next section. We would then discuss content SEO strategy, steps, best practices, and SEO anti-patterns. The next section of the chapter discusses in detail the CMS evaluation framework. We look at key design considerations and detail the comprehensive CMS evaluation framework.

CMS architects, CMS developers, CMS QA team members, and program managers may find this chapter useful.

10.1 CONTENT VALIDATION

Comprehensive content validation tests the content from various dimensions. The content validation methodology covers all real-world scenarios to ensure the delivery of high-quality content.

Let us look at validation techniques and methods used for content validation. The main testing types used for testing the content are:

- **Functionality testing:** This is an elaborate testing that includes testing end-to-end functionality. Some commonly tested content features in this category are:
  - End-to-end authoring flow (from template selection to content publishing)
  - Business rules implementation in the workflows (including multi-level approval flows, workflow routing, etc.)
  - Multi-site publishing and other publishing scenarios (such as Omni-channel publishing, batch publishing, delta publishing etc.)
  - Content-editing scenarios such as in-line editing, content preview, etc.
  - Scenarios of content versioning, archival, asset publishing, etc.
  - Multilingual content handling, translation workflow, and multilingual publishing scenarios.
- **Security testing:** In this category we test for role-based access to content and CMS features such as role-based access to workflows, publishing functionality, administration features, and permission verification for content modules. We would also carry out security code review of CMS components to uncover any potential security issues.
- **Usability testing:** In this category we test the usability of final approved content including the navigation features, interactive features, multi-channel access features, user interface testing, accessibility testing, etc. The aim is to verify if the users can perform intended tasks efficiently and satisfactorily. Some of the key user interface testing scenarios are:
  - Verifying that the content sections are rendered at the intended positions of the page layout

  - Verifying content formatting including the styling, spacing, font specifications, and image position in content
  - Verifying that the links and documents are rendered properly
  - Verifying that the section headers are rendered properly
  - Identifying any user experience issues (such as scroll ability, zooming, font size, color scheme) and readability issues
  - Getting user feedback about the ease of use, navigation model, and information discoverability
  - Verifying the applicability of contextual help
  - Verifying the accessibility on all supported browsers and devices
  - Verifying the performance of the content and presentation.

- **Performance testing:** We check the performance of core content scenarios such as content authoring, workflows, content publishing, content versioning, content creation, page response time, process completion, time and others while handling heavy content volume. In a content scenario, the performance of both authoring environment and publishing environment is tested. During this testing we would monitor various server resources such as CPU utilization, network bandwidth utilization, disk I/O activity, memory usage trends, and database and Web server utilization. During performance testing we may uncover performance bottlenecks and other application performance issues and take necessary steps to address them.

- **Compatibility testing:** We test the compatibility of content on all supported browsers and devices.

- **A/B testing:** In this case variants of content will be used to test the effectiveness on end-users.

- **Localization testing:** In this category we test the translation workflows, locale-specific templates, localized content, etc. Linguistic validation, which validates appropriate usage of the translated text, is also part of this validation.

- **Services testing:** As most of the CMS platforms expose content via services, we also test the content services.

- **Integration testing:** This involves testing the integration scenarios of CMS with other systems such as DAM, metadata system, translation management system, etc.

- **Content flow validation:** In this category we verify the steps of content workflow and all associated business rules against the requirements. Any variations (based on geo or language) are also verified.

- **Infrastructure testing:** The underlying content infrastructure also needs to be tested against the specified nonfunctional requirements (such as scalability, availability, performance).

- **Accessibility testing:** Content is tested against accessibility guidelines (such as W3's Web Content Accessibility Guidelines [WCAG])

- **Load testing:** We test the content system's behavior at expected load. We measure the system's performance and transaction behavior for a specified time period (normally 1-2 hours) and for a predefined set of iterations. In the course of testing we use a mix-load and monitor parameters such as throughput, page performance, response times, and monitor system resources (such as CPU utilization, memory usage, etc.) for all application components.

- **Stress testing:** We load the content system to understand its behavior at upper limits. We load the system with peak load (peak user load, maximum content volume, maximum transactions) and observe the application behavior and system resources. The test measures the behavior of application during peak load and its error-handling mechanism.
- **Endurance testing:** We load the content system with normal load for extended duration (normally 24-48 hours). The test determines system stability under sustained load for long durations. We can determine performance degradation, memory leaks, connection leaks, thread issues, and other issues during this testing.

The objectives and approach for various testing types are summarized in Table 10.1.

Besides these regular validations, more test cases may be required based on specific scenarios. For instance, CMS component testing and content migration testing would be necessary for suitable scenarios.

Content-Testing Checklist

A content validation team can prepare a comprehensive checklist to ensure that various content aspects are covered. Key content testing checkpoints are listed as follows:

- Are all functional scenarios covered in testing plan?
- Does requirement to test a case traceability matrix exist?
- Are pages and content critical to business goals covered in the testing? This list includes content that directly influences customer behavior, such as pages that drive conversions, pages with highest level of user engagement, and pages that contribute to business revenue.
- Does content testing cover popular and frequently accessed pages?
- Does the testing include nonfunctional testing scenarios such as scalability testing, performance testing, accessibility testing, and browser/device compatibility testing?
- Is localization testing covered?
- Does workflow testing cover all user scenarios, exception flow scenarios, and notification scenarios?
- Do test cases cover social and collaboration features such as blog, wiki, communities, and the like?

Table 10.1 Content test types

Test Type	Testing Objectives	Testing Approach
Functional Testing	• To test the content functionality in scope as per the requirement specifications • To test the internal function of the CMS components • To minimize the cost to fix the defects at latter stages • To test behavior of content modules, pages, templates, and workflows and other components as per specifications	• Test team would go through the content requirements and come up with the general positive and negative content scenarios • Test team would prepare the test cases in test management tools based on the identified scenarios • Test execution would be performed on different components, and results would be captured by the test management tool • Defect would be raised accordingly and would be linked with the test cases. The defect would be tracked until the final outcome.
Regression Testing	• To make sure the changed/updated code components are working fine after defect fixes are deployed • Components are behaving as mentioned in the functional specifications • To have confidence before moving to next phase of testing	• Regression test cases would be designed based on the updated content modules • New test cases would be created for new functionality • Regression testing would also be performed during subsequent production releases to ensure backward compatibility
User Experience testing	• To make sure that content is matching the wireframes, copy deck, and visual specifications • To identify the UI issues • To identify any potential issues with page templates, navigation elements and other presentation components • To identify factors for improving performance and user acceptance • To identify potential areas that have maximum impact on the interface's usability • To ensure that world-class experience is provided to the end-users	• Test team would develop the test cases based on wireframes and visual designs. The defect would be raised accordingly for any mismatch. **Sample Tests:** • Testing the image specifications on the page • Testing the navigation model on the page • Measurement of elements in pixels and distance between elements would be measured using ruler • Color value in hex would be measured using color picker

(*continued*)

Table 10.1 (*Continued*)

Test Type	Testing Objectives	Testing Approach
Multi Browser Testing/ Browser Compatibility Testing	• The application would be accessible on all of the specified browsers, keeping the functional and visual aspects in mind • Testing would be performed on all the supported browsers • To verify if all the functionalities are working are per the specification • User experience of the application is same across the browsers	• Testing would cover all pages and page content on supported browsers • The user experience, navigation, and functionality would be tested on all supported browsers • Performance of page and page content would be tested on all supported browsers
Multilingual Testing	• To ensure that content modules are working for all supported languages • To ensure user experience is consistent across all supported languages • To ensure that integration, service calls, and workflows are working across all supported languages.	• Multilingual versions of the content site are accessed through language-specific URLs and functionality testing (specified earlier in the table) is carried out. • Integration scenarios (such as API calls, service calls) would be tested for multilingual content • Input to form fields and content entry fields would be provided in corresponding language and the behavior is verified
Omni-channel Testing	• Verifies that each part of content works according to specification on mobile/tablet devices when all the components are integrated. • To ensure the adaptive design techniques of the content • To test the responsive Web design (RWD) of the content pages	• Content responsiveness when accessed on different devices having different form factors and resolutions. • End-to-end functional testing of the major business flows • Usability of content when accessed on different devices
Performance Testing	• To test the performance, throughput, responsiveness and scalability of the CMS system at various workloads	• The performance test team would come up with the test scripts that are identified as critical business flows. Test scripts would be prepared using performance-testing tools.

(*continued*)

Table 10.1 (*Continued*)

Test Type	Testing Objectives	Testing Approach
	• To thoroughly test various specified performance SLAs such as response time, load times and such at various work loads • To identify any potential performance bottlenecks at high loads • To subject the system to realistic production loads and observe performance • To identify the performance of the applications stress workload in terms of transaction response time • To determine how an application reacts to prolonged duration of load, and endurance test will be conducted	• Average user load and peak user load for the CMS system would be determined and would be used for stress testing and endurance testing • Performance SLAs such as response time, page load time, and asset load time would be verified for content under average load • Performance testing would be carried out on all supported browsers and devices • The performance of CMS integrations would also be tested under average load and peak load • The performance of all exposed content services would be tested under average load and peak load
Security Testing	• To identify security vulnerability at application level, system level and infrastructure level by simulating various security attacks such as injection attacks, XSS, CSRF etc. • To leverage the combination of manual testing steps and automated tools, scripts and frameworks for providing a comprehensive security testing coverage and security reports for the applications.	• Assess the application vulnerabilities • Create security test cases to test various security scenarios such as SQL injection attacks, input validation, XSS attacks, etc. Security testing details are discussed in more detail in Chapter 11.
Load Testing	To subject the CMS servers to realistic production loads and observe performance and system resources	• It is generally executed at the start of test execution cycle • It is executed at peak workloads for 1-2 hours • Gives a measure of the speed, user satisfaction and SLA compliance

(*continued*)

Table 10.1 (*Continued*)

Test Type	Testing Objectives	Testing Approach
Endurance Testing	To determine how a CMS application reacts to prolonged duration of load	• Generally executed after the load test to judge system stability • Executed at normal load conditions for 24-48 hours • Memory leaks, queuing problems, and data sync-up issues would be uncovered during this test
Stress Testing	To subject the CMS system with increasing load until it reaches saturation point	• Executed after an endurance test to determine system's robustness • Executed at 2-3 times peak load condition for 2-4 hours • Helps determine the threshold, robustness, and availability of an application.

- Are content security scenarios (such as XSS, CSRF, injection attacks, DDoS, etc.) covered as part of security testing?
- Are multi-site publishing scenarios covered as part of testing?
- Are content rollback scenario, disaster recovery scenario, and other contingency scenarios tested?
- Is disaster recovery setup verified?
- Does integration testing cover all CMS integration scenarios?

10.2 CONTENT ANALYTICS AND KPIs

Content analytics and KPI metrics are effective tools to continuously track the content usage. We may utilize analytics as a feedback mechanism to experiment and successively increase the appropriateness and positive effects of contextually sensitive content. Analytics form a crucial element of a monitoring infrastructure. In the following section we discuss content analytics in greater detail. We also take a look at important content KPIs.

Content Analytics

Content analytics is all about designing and collecting content usage and user interaction with content. Content analytics plays a vital role in measuring the effectiveness of content and thus sheds insights on how to improve it. The prominent usage scenarios of content analytics are:

- Track usage of content (such as content views, link clicks, time on page, exit rate, bounce rate, etc.)
- Track document downloads (such as number of downloads, most popular downloads, download time, download error)
- Personalize content based on user segment, past viewing/download history, click stream behavior, user roles, user profile attributes, etc.
- Use analytics to measure effectiveness in A/B testing and multivariate testing
- Measure page performance and page load statistics across various geographies
- Analytical reporting, which provides insights into content usage and customer behavior through traffic reports, page view reports, etc.

Content analytics in various content scenarios

In a content management scenario, content analytics can be used to understand user behavior, click streams, and content usage. In digital marketing, analytics can be employed to measure the campaign effectiveness and in e-commerce we can identify product usage, product recommendations, and product-bundling options.

Design and implementation of content analytics

Various phases and associated steps of content analytics strategy arte depicted in Figure 10.1.

Content analytics design starts with identification of the key content metrics that need to be monitored for during the content strategy planning phase (see Chapter 2). Content metrics and KPIs are defined in the content strategy definition phase. The metrics are closely aligned to the business goals and content strategy. For instance, in manufacturing a business domain where the content is mainly used for providing information to end-users, the key business goals are to increase the awareness among customers, to provide consistent visual branding and to enhance information discovery. In order to achieve these business goals, we need to define analytics metrics such as site traffic rate, return visitor rate, time on site, bounce rate, information discovery time, and the like. In an e-commerce domain the main business goal is to increase the sales and to increase the customer base. Therefore, the analytics metrics would be conversion rate, sales per visit, average order value, process completion time, and the like. Key content metrics could also be obtained by baselining existing metrics.

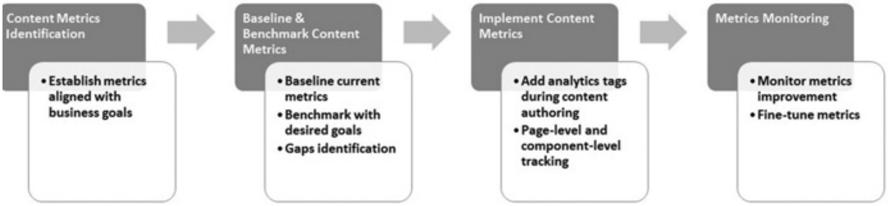

Content Metrics Identification
• Establish metrics aligned with business goals
Baseline & Benchmark Content Metrics
• Baseline current metrics
• Benchmark with desired goals
• Gaps identification
Implement Content Metrics
• Add analytics tags during content authoring
• Page-level and component-level tracking
Metrics Monitoring
• Monitor metrics improvement
• Fine-tune metrics

Figure 10.1 Content Analytics Strategy

Benchmark the baselined metrics against the desired outcomes to understand gaps and influencers. We can then identify the root cause to fill those gaps and formulate metrics around them. For instance, when we benchmark order volume and conversion rate to baselined site traffic metric, we identify the gaps in conversion (like high cart abandonment rates on a shopping cart page, etc.). We can then start tracking exit rate metric, cart abandonment rate, and checkout abandonment rate on the shopping cart page.

Once we have defined the business goals and identified metrics for these goals, we should design and implement the analytics elements to support this. All the content elements (Web content, assets) should be tagged. The most natural choice for analytics tagging is authoring templates. The authoring templates should provide fields to add the analytics scripts and tags. Usually Web analytics needs JavaScripts to be added at the page level and the fine-grained tagging at UI element level (for element-level event tracking). Content-authoring templates should provide fields to add the page-level analytics scripts. UI element-level tagging can be done while authoring content chunks.

Normally the page-level scripts are applicable for all the site pages. Hence we can add them to the global elements such as header or footer for easier maintenance. There are analytics plugins and extensions for most of the CMS products. For instance, Drupal has Google analytics module, piwik integrate module; WordPress has WP-piwik plugin; Joomla has Google analytics plugin and other extensions.

Integration of the Web analytics code is standard for most Web analytics products. We need to insert the main JavaScript file (optionally with unique account number) in the header/footer of each page.

A sample Google Web Analytics script for tracking page-level events is given below. We would need to sign up with Google Analytics service, which provides a Google Analytics account ID (indicated by UA-xxxx-x in the script) required for this script.

```
<script type="text/javascript">
var _gaq = _gaq || [];
_gaq.push(['_setAccount', 'UA-xxxxx-x']);
_gaq.push(['_setDomainName', 'none']);
_gaq.push(['_setAllowLinker', 'true']);
_gaq.push(['_trackPageview']);

(function() {
var ga = document.createElement('script');
ga.type = 'text/javascript'; ga.async = true;
ga.src = ('https:' == document.location.protocol ?
'https://ssl' : 'http://www') + '.google-analytics.com/ga.js';
var s = document.getElementsByTagName('script')[0];
s.parentNode.insertBefore(ga, s);
})();
</script>
```

Note: The Google Web analytics script code given above is for educational purposes only. The exact code may vary based on the software version and requirements.

Once configured, we could track page-level activities such as page visits, navigation top N page visits, most visited/used pages, etc. We need to continuously monitor the analytics reports and fine-tune the metrics as needed.

How can we use content analytics to improve the content strategy?

Let us look at ways in which content analytics can act as a feedback loop to content strategy exercise and influence the content design.

- **Content design and usability:** Insights drawn from the content analytics help the content strategist design highly influential and persuasive content. For instance, if analytics reports and A/B testing indicates that a mix of marquee images along with embedded content is more effective than a normal content chunk, then the content strategist can use the marquee graphics with embedded content for prominent content design.
- **Design an effective return-on-investment (ROI) strategy:** A content strategist can devise the page layout and content placement and effectively use content and graphics to enhance the conversion rate based on the usage metrics obtained from the analytics frameworks.
- **User experience:** We can use the insights gathered from metrics such as site traffic, page views, downloads, and exit rate to enhance user experience and navigation models.
- **Improve campaign effectiveness:** Analyze the click rate and click stream, the results of which can be used for further optimizing campaigns.
- **Analytics reports:** Built-in analytics tools available within CMS will help create a seamless content-tagging experience. Content authors can tag the content chunks and pages with relevant analytics tags to track the content metrics. In addition to this one-time activity, we can provide continuous feedback from analytics reports so that a content author can revisit content and make the necessary changes. If the users are consistently moving away from the Web page, then authors can rework content to make it more usable. Normally we follow these techniques to improve the page content:
  - Summarize the large content chunk into brief summary points
  - Boost the content usability through introduction of How-to content, video-based content, FAQ content
- **Content performance:** Content analytics can also provide insights into the performance of the overall page content. Analytics reports provide details on overall page load time, asset download time, and page response time across geographies. This can help content strategists and authors optimize content through caching techniques and other means.

Case study: Content analytics for an e-commerce site

We will now look at a case study that elaborates various phases in defining content analytics for an e-commerce application. We will look at various content analytics related activities and the tracking metrics starting from "Awareness" phase till "advocacy" phase.

Stage 1: Awareness creation

The primary intent of this stage is to create awareness about the e-commerce brand, products, and offers among potential customers. During this stage, the content marketing team launches marketing campaigns, uses search engine marketing (SEM), e-mail campaigns, and social media marketing, and posts online advertisements to create awareness. The key metrics in this phase are first-time visitors, percentage increase in site traffic, referring sites, average session time, average page views per visit, page response time, most viewed pages, and most frequented paths. Most of these content analytics metrics can be tracked through page-level Web analytics scripts added to the content page. An effective awareness campaign needs well designed Omni-channel content, fast-loading home pages, intuitive information architecture, user-friendly search, navigation model, and other aids for effective information discovery to render simple yet responsive content.

Stage 2: Interest generation

During this stage, we measure content's effectiveness to convert first-time visitors into repeat visitors. Here we need effective content for sustained campaigning and targeted e-mail campaigns, and this should be supported by sustained social media marketing and advertisement campaigns. The key content metrics at this stage are return visitor rate, average time between first and subsequent visits, and referring sites.

Stage 3: User engagement

During this stage, we measure all the user engagement features provided by site content. Content related to user registration, interactive chat, interactive forms, responsive widgets, and collaborative and subscription content plays a key role in actively engaging the users. The metrics tracked at this stage are popular content, most downloaded asset, most page visits, average visit frequency, average session time, and average page views per visit.

Stage 4: Conversion

This is the crucial stage wherein content influences the user's purchasing behavior. Loyalty-related content, product comparison content, discount coupons, promotional content, and targeted and personalized content play a decisive role in increasing the conversion rate. The key metrics in this phase are conversion rate, order value per visit, purchase history, product view history, and click stream.

Stage 5: Advocacy

During this stage customers make repeated purchases with the site and actively participate in discussions, forums, product review/rating, and similar collaborative activities. Customers may also advocate the site to their social media connections. At this stage, loyalty program content and collaborative content play a crucial role in converting a repeat visitor to an advocate. We measure the top activities of the customer, repeated purchases, items most frequently purchased together, category-level best sellers, channel/location specific best sellers, and number of comments, product reviews, and the like. The analytics metrics collected can be used to personalize the user experience and provide most effective product recommendations (based on purchase history and collaborative filtering) and combo offers (product bundling, upselling, and cross-selling). The analytics insights can also be used for providing personalized marketing content and targeted content based on user's affinity.

Another important utility of analytics is identification of customer segments based on their purchase behavior (affinity toward price, brand, demographics, etc.), which can be used to generate personalized recommendations.

Case study: Analytics-driven personalization in Customer Experience Platform (CXM)

Providing responsive and interactive user interface with relevant content is one of the key goals of a CXM. Personalization is one of the main drivers for relevant content delivery. Personalization leverages user's data, past behavior, implicit/explicit user preferences, and other information to provide relevant information for a user/user group. We can use Web analytics software as an effective tool to monitor user behavior. Figure 10.2 depicts an analytics-driven setup for personalization in a CXM platform.

Web analytics can be configured to track user's behavior on the site. It can capture data such as time on page, number of clicks, device, location, preferences, and the like. This behavior data is fed into the "User behavior processor." The marketing team can provide the specifications for user segmentation (based on user type, location, demographics, device, social profile, etc.). The content management system can feed user preferences, profile, historical data, and tagged content to the "User behavior processor." The "User behavior processor" engine then uses holistic user insights to create user segments. User segments can be formed based on user type (guest, reseller, buyer), location of user, device type, and user profile data (gender, income, style, age, etc.).

The recommendation engine can personalize the campaigns, content, and recommendations based on user segments. For instance, the recommendation engine can show the location-based offers (based on user's current location) or trending products (based on user's age segment), etc.

Web analytics continuously monitors the effectiveness of recommendations and forms a feedback loop to continuously improve the quality of recommendations.

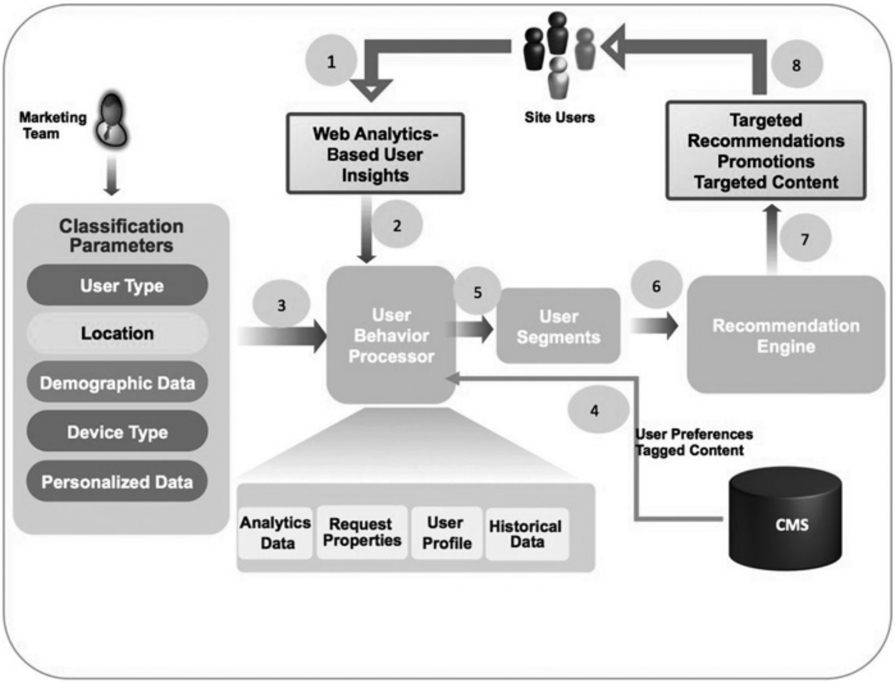


Figure 10.2 Analytics-Driven Personalization for CXP

Content KPIs

Each content program should have clearly defined success metrics. This helps business stakeholders continuously monitor and measure the effectiveness of the CMS system and related processes. Content KPIs can be used to measure critical success factors of the content programs.

Mapping business goals to KPIs

Content KPIs are crucial elements of structured content monitoring framework. KPIs and metrics quantify the content effectiveness, marketing content, campaign content, and others. Each business has its own end goals it wants to derive from content on its site. We need to map those goals to appropriate KPIs. Table 10.2 provides a mapping of business goal, related content, and the tracking metrics for it.

On the CMS platform front, we track metrics such as *content creation cost*, *content translation cost*, *average content publishing time*, and the like to track effectiveness of CMS. The effectiveness of content workflows can be measured using *average process completion time, percent improvement in process response time, percent automation of processes, and similar metrics*. Content reusability is one of the key goals of template and content design. We need to track how content has

Table 10.2 Business goal to content metrics mapping

Business Function/ Business Goal	Related Content	Content KPIs and Metrics
Digital Marketing/ Increased Brand Awareness	Product content Campaign content Social content (co-created by end-users)	Content views Time on site Content shares Number of user comments Asset downloads Video views Document views E-mail open rates Newsletter subscriptions Campaign ROI
Digital Marketing/ Increased Site Traffic	Campaign content Social/collaborative content Self-help content (FAQs, tutorials, tools and resources, etc.) Promotional content Targeted personalized content	Site traffic First time visitors Repeat visitor rate Page views per hour User registrations Contact form submissions Video/page views Unique visitors Time on site
E-commerce/ Increased Conversion	Graphics for effective call-to-action (such as using effective images for checkout button, add-to-cart button, add-to-wish-list button, etc.) Product videos Product comparison content Competitive benchmark content	Conversion rate Visits-to-sales ratio Number of clicks on call-to-action elements Time on cart Cart/checkout abandonment rate
E-commerce/ Increased Customer Loyalty	Targeted content Personalized content Loyalty program content Co-created content (articles, blogs, wikis, forums)	Repeat visitors Upsell/cross sell sales Bounce rate User registration rate E-mail/blog/feed subscription rate
E-commerce/ Increased User Engagement	User-generated content (comments, reviews, feedback, blog, articles) Survey content Poll content	Bounce rate Content likes, social shares Content rating Comments Number of inbound links to blogs/articles Blog/article views Feed subscription rate Social bookmark rate Content co-creation rate

been reused on various pages and sections. For a business user metrics such as *ease of using CMS administration interface, availability of business controls*, and such, self-service-enabling features form main KPIs.

10.3 CONTENT SEO

SEO (Search Engine Optimization) is a set of processes to improve the search rankings of Web content. Adopting SEO best practices enhances the content visibility and drives more user traffic. In a content scenario, SEO can help in having a positive impact on the following factors:

- Increase site traffic due to higher search engine ratings leading to increased awareness
- Increase content visibility and effectiveness
- Increase the return-on-investment (ROI) in digital marketing efforts
- Enhance branding
- Increase online conversion rate for sales content through increased click-through rates (CTR)

Content SEO Strategy

SEO strategy is developed as part of overall content strategy. The business goals and end-user needs, which we gather during the content strategy definition phase, provide inputs for defining content SEO.

The general process steps in defining the content SEO are:

- **Understanding business strategy:** The first step is to understand the general business strategy. We may gain crucial insights such as key business metrics, target audience, user segments, target geos, business-critical pages and content, revenue-generating page flows, target market, and competitors. This step would help us identify business-critical pages and site sections where we can add SEO elements (such as keywords, metatags, etc.).
- **Defining SEO tags:** In this step we define the SEO elements such as high-level keywords, metatags, and SEO-friendly URL structure, page titles, friendly URL, sitemaps, and the like. Some of these activities are overlapping with the information architecture activities. SEO elements such as keywords and metatags are stored in the centralized taxonomy.
- **Specifying SEO tags:** Authoring templates should provide fields for specifying SEO tags. During content-authoring and editing stages, authors can use the centralized taxonomy to reuse one of the existing SEO elements. Additionally they can add other relevant keywords and meta-tags. Quality inbound links to the site's content are a crucial influencing factor for search engine ranking. The

marketing team can provide a link to the site's content from external social channels and from other high-ranking marketing pages and campaign pages.

- **Monitoring and continuous optimization:** Once content is published, the SEO team should continuously monitor the SEO metrics such as search ranking, site traffic, conversions, etc. The SEO process should be continuously improved based on the metrics. SEO monitoring includes page rank analysis, competitive analysis, link analysis, site traffic analysis, and keyword statistics.

In addition to these SEO steps, the marketing team can also actively promote content through marketing campaigns, search engine marketing (SEM), by purchasing search engine advertisement spots (for specific keywords), submitting the sitemap and gateway URLs to search engines, and using paid search.

Content SEO Best Practices

Listed below are SEO best practices, which are applicable in content scenarios:

- Ensure that all content-authoring templates (page-level and chunk-level) have provisions for SEO tags. Authors need to provide relevant keywords applicable for the content. The main SEO tags are:
  - **Title:** Page title should be meaningful and relevant to the page's content. It should preferably contain some of the relevant keywords. Most of the search engines give high relevance to the page title in ordering the search results.
  - **Meta-tags:** We can specify a variety of meta-tags that can accurately describe Web site content. Here are some examples:
    - <meta name="geo" content="US, EMEA, APAC"/>
    - <meta name="targetaudience" content="reseller, customer"/>
  - **Keywords:** Search engines use the keyword value for matching content to the query:
    - <meta name="keywords" content="product overview, product comparison"/>
  - **Description:** This tag provides the brief description of the page's content:
    - <meta name="description" content="This page provides overview of the product and compares the product features"/>
  - **Robots tag:** We can use this to specify the indexing rules to the search engines:
    - <meta name="robots" content="index, follow" /> instructs the search engine to index current page content and follow the page links. We can use this for home pages, seed list pages, and landing pages.
    - <meta name="robots" content="noindex, follow" /> instructs the search engine not to index current page content but to follow the page links. We can use this for site map page.
    - <meta name="robots" content="index, nofollow" /> instructs the search engine to index the current page content but not to follow the page links

    - ■ <meta name=”robots” content=”noindex, nofollow” /> instructs the search engine not to index the current page content and not to follow the page links. We can use this for pages carrying sensitive or confidential information.
  - **Header (H1, H2, H3) tags:** Similar to the title tag, the main heading and subheadings can use meaningful and keyword-based content.
  - **Asset description tags:** Digital assets such as images, animations, and videos are in binary format, which is why the crawling engines cannot index its content as-is. The asset name, image alt tags, and video description/transcript will be used for indexing, and hence they should contain the accurate and meaningful description.
  - **Robots.txt file:** This file specifies the crawalable sections of the site. For instance, the following specification allows all user agents and excludes “/personal/” URL pattern from crawling

    *User-agent: **

    Disallow: /personal/

- Authors can provide all applicable metatag values such as keywords, publication date, author, and so forth. Density of keywords in content will be considered during search ranking.
- Keyword placement, density, and proximity also play key roles in deciding the search ranking. A URL consisting of the keywords, and keywords occurring in close proximity to each other, is a preferred approach. Core content should also contain the specified keywords in close proximity to be more effective.
- If content is rendered as HTML, then the validity of the HTML should be checked to ensure that HTML content complies with the specified HTML standards.
- The marketing team can provide quality in-bound links to the content from trusted sources. High-quality and popular campaign pages and external sources may enhance the search ranking of Web content. A link to main content from promotional and external marketing content, press releases, affiliate sites, feeds, podcasts, advertisements, and external social media platforms should be set up.
- Information architects should create a well-designed sitemap for the site's content.
- For dynamic content, we need to provide the seed list page or quick links page, which can help the search engines reach and index dynamic content. The seed list page should be accessible from the home page or should be specified in the sitemap file so that search engines can follow that link.
- We can leverage robots.txt to specify the URLs to be followed, as well as others.
- To the extent possible, all page URLs should be SEO friendly. They should reflect the navigation structure and content hierarchy and should be easily bookmarkable.

- The page content size should be preferably within 150KB for improved performance.
- Social channels should be leveraged to the maximum extent to promote content. Providing the review and rating options for content, as well as blogs and wikis to end-users, is essential.
- All broken links must be fixed to minimize URL redirects.
- SEO blockers (such as unfriendly URLs, page redirects, etc.) should be identified and steps should be taken to address them.
- We need to ensure that all pages are reachable from the home page, landing page, or through the menus and sitemap. This ensures that search engine spiders can index entire Web site content.
- The sitemap XML should provide links to all pages for a given Web site. It can also provide additional information such as update time and update frequency, which can be optimally used by search engines.
- Search engines would also rank content based on its mobile friendliness. For instance, if content is designed for both desktop and mobile browsers, it should adopt responsive Web design and similar techniques for optimal Omni-channel rendering.

10.4 CMS EVALUATION FRAMEWORK

Selecting the right CMS product is necessary to build and sustain a robust content platform. It is one of the key activities in the CMS implementation approach. In this section we look at factors for selecting a CMS product. We detail a CMS product evaluation framework based on key content requirements. A comprehensive CMS evaluation template can be found in Appendix C.

Business Considerations for Selecting a CMS Product

Some of the key business considerations for a CMS product are as follows. These criteria can be used to select the candidate list of CMS products.

Alignment with business goals: The evaluation framework should check whether the CMS product is closely aligned with short-term tactical and long-term strategic business goals. Some of the important business goals are ease of development and deployment, promoting a self-service model, active user engagement, faster time to market, faster market penetration, easier integration with an existing technology ecosystem, enhancing information discovery, content effectiveness tracking, easier customization and extension, localization support, and multi-site deployment. We need to evaluate the support provided by CMS to realize these business goals.

Total cost of ownership (TCO): This includes license costs, development/customization cost, maintenance cost, upgrade cost, support cost, and hardware/

hosting costs, among others. This helps evaluate considerations such as build vs. buy decisions and consider open-source alternatives and cloud-based products.

Product road map alignment: The evaluation framework should consider if the long-term product strategy and road map are aligned with the business vision for the future. During product evaluation we also look at how well the organization's digital transformation program is aligned with the CMS road map. This could be in terms of feature enablement, enhancement request pipeline, product support, and support for standards and capabilities. For instance, the organization wants to expand its online presence to other geographies; in such cases, a CMS platform should support multiple languages and translation workflows.

Future-readiness: CMS should be able to support forward-looking features such as exposing content as service, adaptive content, Omni-channel optimization, lightweight model, flexible CMS components, social and collaboration support, feed support, agility, cloud support, robust extension framework, and services-based integration.

Functionality fitment: This requires a detailed and multidimensional evaluation of CMS product features with business functionality. Evaluation would happen on various fronts such as core CMS features, fitment to business requirements, operational requirements, and so forth. This will be discussed in detail in the next section. During functionality fitment we also evaluate the list of business requirements that can be implemented using out-of-box (OOB) features. If a good number of business requirements can be implemented through OOB features, this will improve productivity, delivery quality, and time to market.

Publishing capabilities: Content publishing being one of the core requirements for enterprises, we evaluate the publishing capabilities of a CMS product. Ability to do remote publishing, support for publishing in various formats, support for batch publishing, and delta publishing are some of the capabilities we evaluate while deciding on a CMS product.

User experience: We should consider the ease of creating engaging user experience, consistent layout, and uniform experience as one of the key business requirements.

Targeted content delivery: The marketing function may be interested in delivering highly targeted and personalized content to create maximum impact. Content should be relevant, useful, and act as a decision aid and influence users' behavior. CMS should support personalization and visibility rules to support this.

Ease of use: Intuitive user interface, process simplicity, rich component support, out-of-box features, and easy-to-use administration interfaces are some of the factors that make CMS easy to use. These business-friendly features are one of the main CMS evaluation criteria.

Ease of customization: We will check the ease with which it is possible to customize core CMS components, workflows, and deployment processes to suit the business needs in this category.

Extensibility: The CMS should support plugins, adaptors, and extensions to extend the out-of-box functionality. Extensibility and scalability are important criteria to build a robust content platform for future initiatives.

Migration support: If there are content migration requirements, we will also evaluate the ease of migration and support provided by CMS for migration. CMS may support content migration through content export/import, migration tool, bulk content import, and similar features.

Availability of skills: In order to quickly ramp up the team, we need to have people with appropriate skills available in the market.

Market share and installed user base: Business would also be interested in understanding the market share, analyst rankings, and installed user base for the CMS product. Suitability of the CMS for the business-functional domain and reference case studies for the solution domain would also be considered during business evaluation.

Once we have listed candidate CMS products using the business considerations, we can apply the CMS evaluation framework to finalize and select the most suitable CMS product. We have detailed this in the evaluation framework in the next section.

Note: Business considerations may vary based on the business domain. For instance, campaign management business function would have specific business requirements around campaign content management; digital marketing business function would be interested in CMS capabilities to support marketing initiatives; and an e-commerce domain would need support for product catalog and user registration forms. So we need to consider any business-specific needs for the solution domain.

Evaluation Framework

The CMS evaluation framework proposed here provides comprehensive evaluation guidelines for CMS products from four dimensions: core CMS capabilities supported, functionality fitment, technology evaluation and fitment, and operation capability evaluation. Each of these four dimensions has relevant evaluation criteria.

Note: In the CMS evaluation template detailed in Appendix C, we have merged core CMS capabilities as part of functional capabilities for simplification.

This evaluation framework has been successfully used in CMS consulting engagements for evaluation and selection of CMS products.

Using the evaluation framework

These are the general steps for using the CMS evaluation framework:

- To begin the evaluation, we first need to compile the business-critical content requirements, which we would use for the functionality fitment dimension.
- For each of the dimensions, we need to assign the weightage (based on business priority and applicability to solution domain). Normally we assign weightage of 20% to core CMS capabilities, 30% to functionality fitment dimension, 30% to technical capabilities and fitment dimension, and 20% to operational capabilities dimension. The weightages need to be fine-tuned based on business priorities.

Table 10.3 CMS Evaluation Criteria

Evaluation Dimension	Evaluation Criteria	Factors Considered for Evaluation
Core CMS capabilities	Content Authoring	Ease of authoring, availability of RTE editors, authoring interface, availability of content tagging and content validation features, availability of built-in templates, ease of creating and customizing templates, multilingual content editing, ease of creating the components, support for SEO and Web analytics tagging, support for content categories and secured content folder, support for content standards, localization support, support for adaptive content To evaluate the content created from CMS, we can use parameters such as relevance, credibility, accuracy, persuasion, and presentation consistency
	Content Preview and Publishing	Availability of built-in publishing workflows, support for role-based publishing, support for content preview feature, support for in-line and in-context editing and preview, support for presentation templates and style sheets, support for publishing in multiple formats such as HTML, PDF, JSON, XML, XHTML, and DITA, support for distributed publishing
	Content Workflows	Ability to support business processes through workflows, support for scheduled workflows and built-in workflow libraries, support for workflow notification feature, support for workflow design tools, support for multi-level approvals, support for workflow rules, support for rule-based routing, branching support, support for parallel execution
	Content Versioning, Locking, and Archival, Content Classification	Availability of built-in content versioning and archival feature, support for labeling and rollback features, support for version comparison, support for content locking/unlocking for concurrent access, content check-in/checkout, support for content categorization and folder structure
	Content Presentation	Availability of built-in presentation templates, support for user experience management features such as RWD support/Web forms, support for adaptive design, support for content personalization and user segmentation, support for targeted content delivery. Presentation features should support easy integration with visual designs, accessibility standards, branding standards, flexible layout structures, and proper error handling.
	Site Management	Support for site creation, support for friendly URLs and static/dynamic URLs, support for multi-site management and multilingual sites, support for defining and modifying the site hierarchy, support for micro-sites, support for site-based branding and themes
	Taxonomy, Metadata and Tag Management	Support for defining and storing content terms and content categories, support for metadata hierarchy, support for multidimensional hierarchy, support for integration with centralized enterprise-wide metadata, support for site taxonomy

	Document Management	Support for capturing various document formats, support for integration with various document repositories, support for document editing and tagging, support for metadata tagging for documents
	Content Localization	Support for multilingual content authoring, availability of built-in translation system, support for integration with external TMS, UTF-8 support and multi-byte character set support, support for content fallback, support for document localization, availability of language packs and resource bundles
	Content Lifecycle Management	Support for content acquisition, content check-in, content purging and expiration, support for scheduled content maintenance activities
	Security Features	Built-in integration with user repository, support for access-based controls and role-based content access and folder access, single sign-on (SSO) support, support for security flows such as user registration and user provisioning, support for measures to prevent security attacks (injection attacks, cross-site scripting, denial of service, cross-site request forgery, etc.), authentication and authorization controls,oOut-of-box availability of security filters
	Digital Asset Management Features	Ability to upload asset, support for rich media formats such as multimedia files, images, etc., support for asset search, support for context-based asset rendering (such as device-based, preference-based), support for asset workflows, support for asset transcoding, asset versioning, support for changing image resolutions, support for asset editing, support for asset renditions, support for batch and offline asset processes, support for asset tagging, support for acquiring assets from third-party systems, ability to scale and handle large amounts of assets, support for asset caching, integration support with DAM and asset management systems through APIs and services
Functionality Fitment	Business Friendliness and Controls	Availability of business-friendly admin interfaces, availability of self-service features, easy-to-configure interfaces, support for configuration-driven functionality
	Multilingual and Multi-Site Capabilities	Support for multiple languages, multi-site publishing features, support for geo/language specific URLs
	User Engagement Features	Ability to support content for collaborative features such as blog, forums, wiki, rating, polls, surveys, and communities, support for integration with social feed, support for exposing content feeds, ability to provide knowledge management features, support for content syndication, support for integration with external social media platforms

(*continued*)

Table 10.3 (*Continued*)

Evaluation Dimension	Evaluation Criteria	Factors Considered for Evaluation
	Ease of Migration	Ease of migration from existing content sources, support for third-party import tools and ETL tools, support for exporting content to standard formats (such as XML) suitable for migration
	Business Function-Specific Requirements	Support for content personalization, content targeting, campaign management, content marketing, and brand management
	Reporting, Analytics, and SEO	Support for analytics, reporting, and content auditing, support for basic site usage analytics tracking (such as popular pages, page visits, content usage) and reporting, support for page-level and component-level Web analytics tagging, support for SEO features such as friendly URLs/sitemaps/robots.txt/page metadata, support for other SEO features such as page-level keywords, metadata, etc., ease of integration with external Web analytics tools
	Omni-Channel Content	Ability to create adaptive content for multiple devices and browsers, support for responsive Web design (RWD) for page layouts, support for widgets and client-side components, support for content preview on various devices, support for device detection and rendition based on device capabilities, support for device rules, support for responsive themes and mobile themes
Technology Capabilities	Integrations	Support for built-in integrators to security systems, support for integration with portals, availability of out-of-box adaptors for ERP systems, support for content APIs for external consumption, support for integration with translation system, support for integration with metadata system, support for analytics integration, support for integration with DAM systems, support for integration with external social media platforms, support for integration with external content providers (such as external video sites, content services, etc.), support for standards such as JCR and CMIS, support for content services
	Structured Content Support	Support of creation and distribution of structured content in open standards (XML, JSON, DITA, SCORM, etc.)
	Metadata Support	Provision of basic built-in support for semantic metadata. Support for built-in tagging interface, support for integration with external metadata management systems (MMS)
	Support for Standards	Support for content standards such as JSR 170 (Java Content Repository), JSR 283, JSR 168, JSR 286, WebDAV, metadata standards (such as Dublin core), service standards (such as REST, SOAP), data exchange standards (such as XML, DITA, JSON), feed standards (such as RSS and ATOM)

	Flexible Product Architecture	Support for extensions and plugins, support for modular components
	Services-Based Architecture Support	Ability to expose and consume services, support for service monitoring and performance management, support for producing and consuming Web services (SOAP-based and REST-based)
	Performance and Scalability	Ability of the platform to perform within SLAs at high load, support for fault tolerance, support for content (page-level and fragment-level) caching, support for performance optimization features, clustering support
	Availability	Support for disaster recovery, support for cloud hosting, support for content failover, support for load balancing
	Content Search	Support for searching and indexing all types of content, ability to integrate with external search engines, availability of connectors to search engines, availability of built-in content search engines
	Personalization Support	Support for segmenting users based on user profile attributes, demographic information, context information, etc., ability to customize the content dynamically based on user information, support for A/B testing and multivariate testing, ability to personalize content based on user profile attributes and content metadata
Operational Capabilities	Flexible Hosting Models	Ability to support cloud and in-house hosting models
	Ease of Maintenance, Support and Adoption	Ease of implementing enhancements, faster time to market, support for automated patching/upgrades and maintenance activities, support for automated testing, community support, product documentation, availability of skillset in the market, availability of training services, availability of product training and documentation
	Ease of Administration	Support for content migration, support for managing roles and permissions, support for managing workflows, support for configuring search parameters, availability of out-of-box adaptors with DAM, document repositories, and other required interfaces
	Release Management	Support for easier packaging, support for code deployment across environments, support for partial and delta deployment, support for integration with external deployment tools, multi-site deployment, clustered deployment

- It is essential to evaluate the fitment of each of the candidate CMS products against the evaluation criteria in each of four dimensions. We map the CMS solution components to the evaluation criteria during this process. The level of fitment can be "fully compliant" (if CMS satisfies the requirement out of the box), "partially compliant" (if the requirement can be realized through CMS configuration changes or through CMS extensions), or "not compliant" (CMS product does not support the requirement and we need to develop a custom component or augment another product for filling the gap).
- We use the aggregated weighted score (dimension weightage x fitment score) to arrive at the final score and finalize the CMS product.

Note: In addition to this evaluation exercise,e sometimes other criteria will also be considered. This includes proof-of-concept (PoC) outcomes, existing content ecosystem, existing investment, alliance partnership,s etc.

Multidimensional evaluation criteria

Let us look at the evaluation factors in each of the dimensions. Table 10.3 lists the key evaluation criteria under each dimension.

10.5 APPENDIX: WCMS FEATURES

In this section we look at various features offered by some of the popular Web CMS products. This is intended to provide the idea of popular features offered by the leading and open-source CMS products.

Note: The feature and product list is partial and is for educational purposes only. It is intended to bring awareness about the WCM products and its features. It is not a comparison, ranking, or recommendation of any product or technology.

Feature	Liferay 6.2 EE	dotCMS 3.2	Drupal 7	Joomla 3.4
Page Template [Creation, Administration and Templating]	Liferay themes and page templates. Support for Velocity and FreeMarker templates Out-of-box page and content administration	We can leverage starter site with predefined templates along with themes Support for custom layouts and WYSIWYG template designer Out-of-box page and content administration	Built-in templates Fluid and dynamic page layouts Administration panel for assigning layouts and administration	Built-in templates Out-of-box administration area

Feature	Liferay 6.2 EE	dotCMS 3.2	Drupal 7	Joomla 3.4
Page Section/ Widget Support	Liferay portal provides support for portlets and AlloyUI-based components Dynamic drag-and-drop support	Supports widgets with help of containers	Supports widgets	Supports widgets
Content and Asset Management	Support for Web content structure and templates, metadata, dynamic tagging and live page editing Provides built-in permission model Supports Web content publishing, asset publishing, unified content, and digital media Provides web content display portlet, asset publisher portlet for content management	Supports content and asset management through features such as asset storage Provides role-based categories and taxonomy permissions and content versioning	Supports content categories, versioning, and URL aliasing Supports document management through modules (such as document and filedepot) Supports role-based permissions Supports multiple modules to manage assets (such as Asset, Asset Manager, Audio, Media etc.)	Supports user roles and permissions, content types, and category systems Supports documents and asset management features through various extensions (such as JoomDOC, DOCMan, etc.)
CMS Integration Capability	Web services (XML-RPC/ SOAP/REST)	Web services (XML-RPC/ SOAP/REST)	Web services (XML-RPC/ SOAP/REST)	Web services (XML-RPC/ SOAP/REST)
User Management (add/update/ delete users, assign user permissions)	Provides built-in user and role management and robust permission model	Provides built-in user management features to manage users and roles and supports permissions	Provides built-in user and role management	Provides built-in user management features to manage users and groups

Feature	Liferay 6.2 EE	dotCMS 3.2	Drupal 7	Joomla 3.4
Performance	Liferay supports various caching frameworks (OSCache, EhCache), which can be leveraged	The enterprise edition supports many performance features such as block cache support, page cache support, clustering, etc.	We can leverage APC (Alternative PHP Cache), Memcache (for caching database content), Varnish (for caching presentation elements), views cache, block cache, entity cache, display cache, and other caching features along with a reverse proxy for optimal Drupal performance	Page caching, view caching, and module caching can be leveraged for caching Additionally, gzip compression and image compression can be used for further optimizations
Taxonomy Support	Supports tag cloud, folksonomy, and taxonomy features	Built-in support through tags, tag cloud, categories, and relationships	Supports taxonomy features	Provides built-in tagging system and taxonomy extensions that can be leveraged in Joomla
Standards Supported	JSR 170, JSR 168, JSR 286, JSR 220, WSRP	Supports JSR 168, CMIS	WSRP is supported through the WSRP module	
Collaboration Support	Built-in support for a collaboration suite, which includes blogs, message boards, forums, community, e-mail, calendar, feeds, wikis, polls, tagging, and announcements	Supports feeds, blogs, wikis, calendar, forums, tagging, etc.	Community modules available for blogs, forums, and wiki Collaboration features are also supported through modules (such as forum, comment)	Collaborate extension can be leveraged

10.6 CHAPTER SUMMARY

- Content validation includes functionality testing, security testing, performance testing, compatibility testing, A/B testing, localization testing, services testing, integration testing, content flow validation, infrastructure testing, accessibility testing, component testing, and migration testing.
- Content analytics and KPIs are mainly used to monitor and track content usage.
- Content analytics are designed based on content and business goals. We need to add the corresponding tags to the content templates and elements.
- Content analytics can help in improving content usability, content performance, and ROI.
- The chapter discussed a case study for content analytics for an e-commerce site. The main steps are awareness creation, interest generation, user engagement, conversion, and advocacy.
- Content KPIs are effective measurements of success of content programs. KPIs are mainly based on the business function and business goals. A mapping of business goals to the content KPIs has been provided.
- SEO is used to enhance content visibility. SEO provides high search engine rankings, ROI, and conversion rate.
- Main steps in defining SEO are understanding business strategy, defining SEO tags, specifying SEO tags, and monitoring and continuous optimization.
- Content governance defines the processes, roles, and responsibilities for managing content.
- The main drivers of content governance are uniform and consistent branding, defining standards, well-defined processes for handling content operations, increased productivity, enhanced content quality, enhanced user experience and active engagement, and satisfying business SLAs.
- The main PMO roles are program sponsor, program director, content architect, and content manager.
- Various program management activities and deliverable for various project lifecycle phases have been explored.
- Main business considerations for selecting a CMS product are business goals alignment, total cost of ownership, product roadmap alignment, future readiness, functional fitment, publishing capabilities, user experience, targeted content delivery, ease of use, ease of customization, extensibility, migration support, skill availability, market share and installed user base.
- CMS evaluation framework provides evaluation criteria on four dimensions: CMS core capabilities, functionality fitment, technical capabilities and operational capabilities.
- CMS core capability dimension includes content authoring, content preview, content workflows, content versioning, content presentation, site management,

taxonomy & metadata support, document management, localization, lifecycle management, security, DAM support.

- Functionality fitment dimension includes business friendly controls, multilingual capabilities, user engagement features, ease of migration, business requirement fitment, reporting, analysis, Omni-channel content.
- Technology capabilities include integration, structured content support, metadata support, standards support, services based architecture support, performance, scalability, availability, content search, personalization support.
- Operational capabilities include flexible hosting models, ease of maintenance, support and adoption, ease of administration, release management.

Chapter 11

Content Security

As content management systems are widely used for building public-facing platforms, they are naturally exposed to a wide variety of security threats. Site defacing, information theft, injection attacks, and scripting attacks are some of the most common threats faced by popular CMS portals. A compromised public site may severely impact the credibility and reputation of the organization. Comprehensive and robust security is essential to effectively mitigating the security risks.

In this chapter we take a look at various content security aspects. We look at the details of achieving course-grained and fine-grained security measures and content-related security policies. To begin with, we look at common security vulnerabilities such as Cross-site scripting (XSS), injection attacks, denial-of-service attacks, CSRF, and click-jacking along with measures to prevent them. We then look at core content security concerns such as authentication, authorization, SSO, permission model, and security testing scenarios. In the section that follows we look at various security best practices such as layer-wise security, account management, transport-level security CMS hardening, and others. The chapter concludes with a discussion of a security-testing case study for a CMS application.

Content architects, security architects, enterprise architects, and CMS developers will find this chapter useful.

Note: Although the chapter provides examples for open-source systems and Java-based application servers, the generic concepts remain valid for other platforms as well. Wherever possible, we have tried to provide examples of non–open source systems as well.

11.1 CONTENT SECURITY VULNERABILITIES AND MITIGATION STEPS

In this section we discuss general security threats faced by the CMS and the best-practice measures to mitigate those security risks.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

Cross-Site Scripting (XSS)

XSS is one of the most common security threats faced by online applications. In XSS cases, the attacker executes the malicious script to steal the security credentials such as session details and other user personal information. In a non-persistent XSS scenario, an attacker adds malicious code as a payload, such as the following: http://www.example.com/home.jsp?p1=“><script>alert(‘test’)</script. The attack normally involves injecting the malicious script in the URL parameter. The malicious payload can load the script from a different domain and can send the session details such as session ID, cookie details, and others to the target domain. The attacker can also introduce the malicious code when entering user-generated content (UGC) such as comments, blog posts, and reviews. As this will be rendered to multiple users, the impact level in case of persistent XSS will be high.

Preventing XSS

CMS can adopt these techniques to prevent the XSS:

Design robust content security policy for resources We can use the directives to the user agents (browsers) to specify the resource-loading behavior. W3C-compliant HTTP headers or meta-tags such as Content-Security-Policy, X-Content-Security-Policy, and X-WebKit-CSP can be used along with directives such as *default-src* (to specify default resource loading policy), *script-src* (specifies the scripts that can be executed), *img-src* (to specify image loading policy), and the like. W3c specification provides details on content security policy at http://www.w3.org/TR/CSP.

Some of the popular CMS products offer plugin support to implement this technique. WordPress provides a “content security policy” plugin; Drupal provides SecKit that implements various aspects of content security policy. In addition to the built-in CMS support for content security policy, we can also configure controls at the Web server level. A sample configuration for an Apache Web server is presented below. These configurations can be added to httpd.conf file or .htaccess file

```
<FilesMatch "\.html>
    # Only allow same origin pages to be framed and deny other
origin others
    Header set X-Frame-Options "SAMEORIGIN"
    # Enable XSS protection
    Header set X-XSS-Protection "1;mode=block" env=ie
    # Content Security Policy specification
    Header set X-Content-Security-Policy "default-src ’self’;
img-src ’self’; style-src ’self’ ’unsafe-inline’; script-src ’self’
https://example.com ; font-src ’self’ ;"
     Header set set X-Permitted-Cross-Domain-Policies "none"
</FilesMatch>
```

In the above configuration, the content security policy specification restricts the image source, style source, and font source to "self"-indicating that resources will be loaded from same origin. *script-src* value specifies that scripts loaded from the same origin or from example.com are trusted. We can also specify other content security policy directives such as *connect-src* (to specify the XHR and websockets connection origins), *media-src* (to specify the audio/video origins), *child-src* (to specify the URLs for embedded frames), and *form-action* (to specify valid form action targets).

Some CMS platforms use other Web servers such as Microsoft IIS platform. In such cases we can use the Web.config file of the IIS server. A sample setting for *default-src* is given below.

```
<system.webServer>
  <httpProtocol>
    <customHeaders>
      <add name="Content-Security-Policy" value="default-src 'self';"/>
    </customHeaders>
  </httpProtocol>
</system.webServer>
```

Leverage OWASP's AntiSamy library AntiSamy library (https://www.owasp.org/index.php/Category:OWASP_AntiSamy_Project) provides validation and sanitization of the HTML against a set of predefined rules. All blacklisted and special characters will be encoded, preventing the persistent XSS scenario.

Use custom application filters We can also develop custom intercepting filters that process all the input HTML values and URL parameters and validate them against whitelist characters. The filter can then encode or escape HTML special characters and sanitize the payload before sending the data to further layers.

Input validation For user entry fields, we can perform client-side validation to prevent users from entering special characters or malicious code. We can perform similar checks on the server side as well to identify any malicious code. Often a combination of client-side validation and server-side filtering is more effective. User-generated content (such as comments, wiki articles, blogs, reviews, posts, feedback) needs most attention in this regard. Validating the user inputs (including UGC content, URL parameters, URL schemes, uploaded files and such) against the white list is essential, as well as validating the character set of the uploaded files. One of the effective techniques is to specify the user input type and restrict it wherever possible (such as file type, file size, allowed character set, etc.), which helps with efficient filtering at the source.

SQL Injection Attacks

Most of the CMS platforms use a database at the back end for storing user information and other system/application-specific data. Attackers can launch SQL injection attacks to exploit any application vulnerabilities. A popular form of SQL injection attack uses the page parameters to pass the SQL query parameters: http://server/index.php?name=abc OR 1=1. In this scenario the query engine always returns true due to the OR condition. An attacker can get information about any user or object using such technique. A variation of this technique includes SQL comments, SQL escape values, and others.

Preventing SQL injection

Effective ways to prevent the SQL injection attack are:

- Always validate user input. We need to validate the size, length, format, and data type of the user inputs. Create robust filters that remove or encode special characters, escape sequences, known SQL injection patterns, comments, and binary data.
- Always use prepared statements, especially while appending query filter parameters.
- Use database-stored procedures for data-intensive operations.
- Never concatenate users' input directly into SQL queries. Validate the user input parameters.
- At the database side, do not give table-level privileges (such as alter table, drop table) to CMS users. Wherever possible, create a separate database schema for storing application data.
- We can also define rules in the Web server configuration to filter the malicious URL parameters.

Denial of Service (DoS) and Distributed Denial of Service (DDoS)

In a DoS attack, the attacker overloads the CMS by flooding it with requests. As a result, the CMS exhausts all the resources, making the service unavailable. For instance, an attacker can craft a payload to request all content nodes of the content repository; when CMS is flooded with such requests, it exhausts all its resources to serve the request.

DDoS originates from multiple distributed systems and is more challenging to prevent than DoS is.

Detection and prevention of DDoS

Network-level filtering rules Network devices provide traffic-monitoring and filtering rules that can detect anomalies in the traffic patterns. These rules can be configured to detect the DoS attack originating from a particular machine.

Cloud deployment In case of hosted CMS deployment, cloud providers offer many preventive measures against DoS- and DDoS-style attacks (such as load distribution among various nodes).

CMS-level preventive measures Any CMS service or functionality that can potentially overload the system resources (such as listing the content nodes for the entire repository, or serving a huge list of content for a search query and such) should be carefully designed. First, such functionality should be access-restricted to a limited set of user roles; second, there should be a maximum limit to the total resources served per request (we can adopt pagination of content search results instead of dumping all search results in a single page).

Resource caching may reduce the load on origin CMS servers in case of such attacks. If the assets (such as images, video) were cached at the CDN layer, then the request would not go to the origin server, thereby reducing the load.

Cross-Site Request Forgery (CSRF)

In a CSRF attack, the attacker exploits the trust between the Web site and the user agent. The attacker can execute unauthorized user actions on user's behalf. Let us look at some of the variants of CSRF vulnerability:

- **Posting data and executing commands without user's knowledge and authorization**: Once the user is logged in, the Web site establishes a user session and trusts the user's identity (this is commonly done through security cookies and session identifiers). In such scenarios, an attacker can guess a URL pattern and execute action without the user's knowledge. The attacker can provide the link to the command that can cause a side effect in user-generated content (such as blogs, discussions, articles, etc.). When the logged-in user clicks on that link, he/she unknowingly executes that command. Let us look at a simple example: a hyperlink in the solution article contains the following href value:

  ```
  Href=/changepwd?username=abc&pwd=xyz
  ```

  In this example, the attacker has identified that “*changepwd*” is a command exposed to logged-in users and its parameters. Since the Web site trusts the commands coming in from a valid user session, the command gets executed (on behalf of the logged-in user), and the attacker can change the password of any user. CSRF is one of the vulnerabilities posed by UGC.

- **Sensitive information extraction:** Another type of CSRF vulnerability tricks logged-in users to send the user details or session details to the attacker's site. A simple example of this type of vulnerability is given below:

  ```
  <img src=″/includefile=http://example.com″ />
  ```

The attacker can send the secured session details to the example.com and can use it to launch further attacks. Same technique can be used to steal user cookies and other session data.

Prevention of CSRF

We can adopt the following security measures to prevent CSRF.

Application-level validation As a general rule of thumb, CMS should expose admin and system-level actions or commands, such as password change, only to admin users. It is also recommended that these commands should not be exposed over the Internet. Sensitive commands should be protected with multiple levels of authentication (for example, password reset function should re-authenticate the user or force a two-factor authentication). These preventive measures would reduce the attack surface.

Request authorization tokens This is the most effective way to prevent the CSRF attacks. The CMS system can create a unique hash value token for each web request (it could be a combination of user data, action, site identity, etc.), and the CMS validates this token for each request. The system can then validate the token to ensure it is not a forged request. Since the token is generated and validated at the server side, it would be difficult for the attacker to mimic this. The token can be part of request data:

```
<input type=″hidden″ name=″csrftoken″ value=″d8389soleo925″>
```

Though it adds an additional overhead, this token-based approach ensures the integrity of the request and would be critically essential for secured transactions.

Most of the CMS provide built-in protection against CSRF attacks: Drupal CMS prevents this through anti-CSRF tokens; Craft CMS provides *enableCsrfProtection* configuration for enabling the feature.

Clickjacking

In this kind of attack, attacker tricks the user to perform an action unknowingly and invoke unwanted requests and perform unwanted transactions. The user would think that he/she is clicking on a button or an image, but internally it would execute a script or trigger an action crafted by the attacker through a hidden page.

Prevention of click jacking

To prevent this kind of attack, we can set the X-FRAME-OPTIONS HTTP header to SAMEORIGIN in a Web server configuration. A sample configuration is given below for Apache Web Server:

```
<FilesMatch "\.html>
    # Only allow same origin pages to be framed and deny other
origin others
    Header set X-Frame-Options "SAMEORIGIN"
</FilesMatch>
```

In Web.config file of MS IIS server X frame option can be set as follows:

```
<system.webServer>
  <httpProtocol>
    <customHeaders>
      <add name="X-Frame-Options" value="SAMEORIGIN" />
    </customHeaders>
  </httpProtocol>
</system.webServer>
```

We can prevent the iframes altogether in the application through "*X-Frame-Options: deny*" or by using frame breaker scripts.

11.2 GENERIC CONTENT SECURITY SCENARIOS

In this section we take a look at generic content security related scenarios. We discuss the key security concerns for CMS systems.

Authentication and Authorization

Authentication is the process of verifying user credentials. Upon successful authentication, a user session is created. *Authorization is a fine-grained security model in which a user can access the functionality, pages, and content based on the assigned roles and permissions*. Most CMS provide native authentication and may also support integration with external user registry (such as LDAP or a directory service). In case of native authentication, user profile data would be stored in CMS systems. In this setup, CMS would also store the role information, user role mapping, and role permission mapping. For *external authentication*, the user data will be synchronized to CMS from external LDAP for optimizing performance and for reduced latency (this technique is popularly known as "user provisioning"). User provisioning jobs can be used for two-way user data synchronization between CMS and LDAP. If the organization has a proprietary authentication mechanism, *custom authentication* may be developed to integrate with such proprietary authentication. We leverage standards (such as JAAS), APIs, security services, and security extensions for developing custom authentication modules. They may also support a fine-grained authorization model through role-based access and a permission model that provides role-based access to resources. This permission model is discussed in the upcoming sections.

Single Sign-on (SSO)

In an enterprise scenario, CMS would be one of the applications. In a complex enterprise ecosystem a user navigates multiple enterprise systems (such as portals, CMS, reporting system, etc.) in a user journey. As most of the systems are secured, we should avoid multiple user authentications through a single sign-on (SSO). SSO allows all subscribed enterprise applications to use an enterprise-wise authentication token, thereby providing seamless access to authorized functions and data.

SSO is usually implemented by enterprise security managers (such as CA SiteMinder, Oracle Access Manager, IBM Tivoli Access Manager, etc.). A typical SSO flow is given below. CMS should be integrated with enterprise security managers for SSO to work.

- All requests to secured CMS resources (such as page, content, document and such) will be intercepted by security manager plugin (which is normally installed at web server).
- The security manager would check for security tokens (such as user token, SSO token). If tokens are present, they would be validated and user would be given access to authorized resources else user would be redirected to a user login page for authentication. Upon successful authentication, a new user session would be created.
- On authenticating the SSO token, the security manager gives access to the application seamlessly if the user has clearance to access the application.
- If an SSO token does not exist, the user needs to provide the authentication details in the login page. Upon successful authentication, the security manager generates an SSO token.
- The SSO token remains in the browser cookie and enables the user to log in to other subscribed enterprise applications seamlessly. The SSO token may be used by all enterprise applications such as portal, CMS, etc.

CMS supports SSO through various means. A CAS module can be used for Drupal, and we can also use OpenID and OAuth for achieving SSO in Drupal.

Federated SSO involves authentication with external applications. Security Assertion Markup Language (SAML) is the most popular way to achieve federated SSO. SAML SP module makes Drupal a service provider. In WordPress, we can use SAML 2.0 SSO plugin for authenticating use SAML Identity providers.

Permission Model Using Roles and Permissions

A permission model specifies the fine-grained access control to various content resources. Resources, in the context of content management, are essentially content objects (such as templates, workflow, jobs, pages, etc.) and content functions (such as authoring, publishing, administering, search, deleting, etc.). The first step in defining a permission model is to identify all content resources. We then define the permissions

Table 11.1 Basic CMS permission model

Permission	Privilege
Read	Permission allows the user to read the content resource
Update	Permission allows the user to update and modify the content resource
Delete/Purge	Permission allows the user to delete the content resource
Create	Permission allows the user to create the content resource
Lock/unlock	Permission allows the user to lock or unlock the content resource
Archive	Permission allows the user to archive the content resource
Publish	Permission allows the user to publish the content resource

for managing the content resources. For instance, we can define permissions such as *create, update, delete* permissions on a template resource and we can define *Full Access* or *No access* permission on the authoring function.

Let us look at the main permissions from the content standpoint. A basic permission model for a content system is presented in Table 11.1. The first step is to define the permissions applicable for the system.

The content resource could be one of content chunk, content page, content template, workflow, content asset, and the like.

Once we define the permissions, we can then define the role. We can create a role (or a group) by mapping each role to a set of resource permissions. Roles are bundled with permission for a specific resource. Some roles for content authoring flow are defined in Table 11.2. The table is a two-level mapping: of role to permissions and of permissions to resources.

In the table, a content_author role has create content, read content, update content, and lock/unlock content on the authoring template, which means that the role allows the user to create content using the permissions but is not authorized to change templates or workflow. A content admin, on the other hand, has full permissions on content object such as templates and workflow.

Table 11.2 Content roles and permissions

Role	Permissions	Resource
Content_author	Create content, read content, update content, lock/unlock content on content object Use permission on content function	Content object: Authoring template Content function: Authoring
Content_admin	Full permission on content object Full permission on content function	Content object: Authoring and publishing template, workflow Content function: Authoring, publishing, starting/stopping workflow
Content_reviewer	Read content, update content on content object	Content object: Authoring template

The next step is to assign the roles to required users. When users are assigned roles, they automatically inherit the corresponding permissions. Total number of permissions for a given user is the sum total of inherited permissions for all his/her assigned roles.

Each of the CMS users is assigned to a group or a role. The user inherits all the resource permissions by virtue of the assigned role. CMS can maintain its own user repository to maintain the user-role mapping or it can access the user details from a centralized repository.

Roles can be categorized into mainly two types: *system roles* and *application roles*. The roles that have permissions for system operations such as content authoring, publishing, administration, and impersonation are categorized as system roles. The key roles in this category are content author, content editor, content publisher, content administrator, and content workflow administrator.

Application roles are specific to the functional and business domain of the application. For instance, the manufacturing domain contains roles such as consumer, reseller, partner, and the like.

Permission inheritance hierarchy

One of the most effective ways to manage the permissions in a content scenario is through inheritance. Permission inheritance happens at two levels: *role level* and *resource level*. As the system evolves, we will get numerous functionalities requiring a lot of content resources and content functions. As a result, the number of the necessary permissions will also increase, potentially leading to permission governance and management issues. In this section we look at ways to manage permissions efficiently.

Permission inheritance through role hierarchy We can define a role hierarchy for efficiently managing the permissions. For example, we have seen in the permissions table that a lot of resource permissions are repeating across roles. For instance, both "content_author" and "content_reviewer" have permissions to read and update content on the template content resource. As the roles and resources grow, managing this may become challenging. Hence, we can define a role hierarchy to manage this efficiently. We can define the simple hierarchy for the aforementioned permission table as follows:

Content_reviewer (parent) → content_author (child)

Content_author role, being the child of content_reviewer role, inherits all the resource permissions from content_reviewer. In this case content_author inherits read-content and update-content permissions on the template content object. In addition to this, we can add the permissions specific for the content author role (create content, lock/unlock content on the template content object). Along the same lines, we can restrict or override some of the permissions during the inheritance procedure. This may also simplify permission management.

Permission inheritance through resource hierarchy CMS may also define the resource hierarchy mainly for system resources. This defines another level of permission inheritance. In the permission inheritance hierarchy in Figure 11.1, if any role has full permissions on the "folder" resource, that role automatically inherits full permissions on all its child nodes such as pages, files, menu links, and contained content chunks.

We can also selectively grant the permissions on subfolders to applicable roles.

11.3 SECURITY TESTING

We have discussed the main content security vulnerabilities such as XSS, SQL injection, DoS, CSRF, and clickjacking in the previous sections. In those sections we had also given sample code/attack details for each of the security vulnerabilities. The sample attack code can be used for security testing. The key activities in security testing are core security testing, security code reviews, and penetration testing. Let us look at these aspects in the following sections.

Core Security Testing

The main steps in doing content security testing are as follows:

- Conduct white box and black box testing for each release to discover any security issues before production deployment.
- Conduct security tests of all components, plugins, and extensions of third-party components that will be installed on all layers.
- Perform security code reviews and penetration testing of the applications to discover potential security vulnerabilities.

Security Code Reviews

- Develop security coding guidelines, checklist, and best practices that can be used during application development.

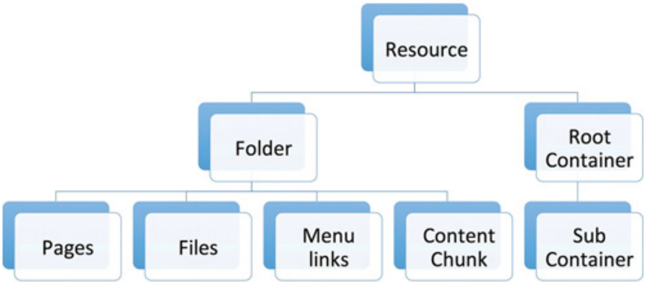


Figure 11.1 CMS Permission Inheritance through Resource Hierarchy

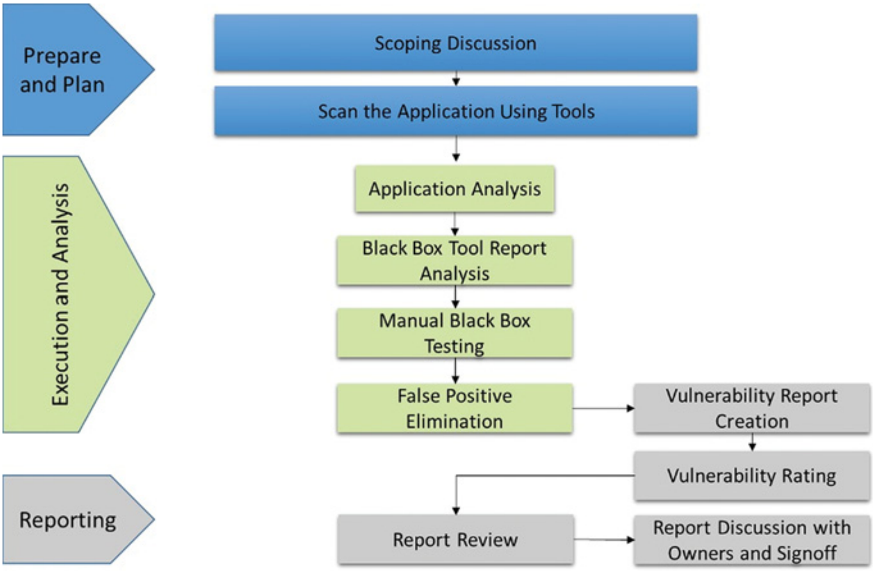

Prepare and Plan
Execution and Analysis
Reporting
Scoping Discussion
Scan the Application Using Tools
Application Analysis
Black Box Tool Report Analysis
Manual Black Box Testing
False Positive Elimination
Vulnerability Report Creation
Vulnerability Rating
Report Review
Report Discussion with Owners and Signoff

Figure 11.2 Penetration Testing Approach

- Conduct regular manual code reviews to identify security coding violations such as hardcoding user credentials, storing secure data in plain text format, and the like.
- Leverage static code analyzers to detect any security coding violations in the code.
- Ensure that no password is stored in clear text in any layer during code review.

Note: We have given sample security checklist in Appendix B, which can be used along with security review.

Penetration Testing

Penetration testing is conducted to exploit the system's vulnerabilities to gain access to system's resources, data, and services.

Sample steps in the penetration testing process are presented in Figure 11.2.

The approach for application vulnerability assessment is as follows:

- Prepare and Plan:
  - Security Scope Definition: Identify the content application vulnerabilities and threats
  - Run the vulnerability scanners on the application to identify known vulnerabilities
  - Review application architecture from the security standpoint
  - Understand the content flows, pages, and navigation models that are vulnerable

- Execution and Analysis:
  - Carry out black box security testing based on the application functionality
  - Compile the identified vulnerabilities and remove the false positives
- Reporting:
  - Analyze the identified vulnerabilities and prioritize them based on their impact potential
  - Report the prioritized list to all stakeholders.

11.4 SECURITY BEST PRACTICES

In this section we look at some of the proven security-related best practices for CMS platforms.

Adopt Multi-Layer Security

- Implement security at all system layers and all application layers. Use secured configurations at Web server, application server, database server, and network layers. At the application layer, use security filters, encryptions, and similar measures. Employ security techniques at all application layers (such as view layer, controller layer, model layer).
- Use only secured access protocols (such as HTTPS, SFTP) to provide transport-level security. Use this for sensitive functions such as file uploads, admin functions, user registration, etc.
- Network-level security: Update the network-level firewall rules and configure rules to detect and prevent known network-level attacks.
- Database-level security: Adopt vendor-recommended security best practices. It is recommended to use separate database schemas to store the system data (required by CMS and its plugins) and application/user-level data (to store user-related content). Also, restrict the write access for system schema only to a database administrator who can monitor the installation of database objects in this schema. Regularly back up the system and application data to remote data centers.
- Web server–level security: Update the Web server configuration files to restrict user access and enforce file permissions and security policies. Block access to sensitive server configuration files and disable directory listings. Additional security filters, plugins, and firewalls can be used along with Web server.
- Install and run antivirus and vulnerability scanners on all servers and schedule a periodic scan job on all systems in a content ecosystem.
- Asset and document repository: Use fine-grained access based on user roles to provide restricted access for digital asset and document management.

Robust Account Management

There should be comprehensive user account management policies that include:

- Strong password policies that require a password string to have a combination of letters, numbers, and special characters
- Avoiding dictionary words for passwords
- Default admin password should be changed immediately after CMS installation
- Adopt periodic password change policy and force users to change the default passwords
- Store the passwords in encrypted format in the CMS database; use one-time hash value (generated via secure hashing algorithms such as BCrypt Password Encoder) along with random salt values while storing the passwords.
- Apply two-factor authentication for critical functionality such as profile updates, password updates, user impersonation, and user creation
- Support for administration interface should be only through a secured transport layer; restrict access to an administration interface
- Use CAPTCHA for sensitive functionality such as password reset, user registration, form submissions, etc.
- Thoroughly verify the SSO logout functionality to ensure that each user session is completely logged out
- Adopt security best practices in session management; use “HttpOnly” attribute for session cookies, properly configured session timeouts, and avoid multiple simultaneous session

Proactive Scanning and Vulnerability Assessment

- Use vulnerability scanners to proactively identify the CMS vulnerabilities. For instance, WPScan for WordPress vulnerability scanner and the *Security review* module for Drupal can be used for security scanning.
- Use other static security code assessment tools to regularly check for security vulnerabilities in the code
- Install the vulnerability scanners on all CMS servers and run them periodically
- Include vulnerability scanning as part of CMS governance process along with patching and updates

CMS Patching and Upgrades

- Establish a governance structure for applying regular patches and upgrades for all of the CMS infrastructure components. We need to consider patching

OS, Web server, CMS (authoring and publishing instance), CMS plugins, and custom code on a regular basis.

- Regularly and in a timely fashion install the vendor-released security patches applicable for the CMS

Transport-Level Security

- Content in motion needs to be encrypted at various levels by applying SSL certificates at all the required levels.
- Content may be transmitted through encrypted HTTPS protocol; use the *Strict-Transport-Security* HTTP header.
- SSL token would be used for API call authentication.
- Secure restful content API for content services.
- Use “secure” cookies for storing sensitive information.
- Use secured protocols such as SFTP and SSL/TLS for data transport.
- All sensitive operations (such as user authentication, user registration, viewing user profile page, etc.) should happen only over HTTPS.

CMS Hardening

- Harden the CMS hardware by restricting ports and protocols. Disable auto execution of files.
- Scan all the files uploaded by end-users and restrict uploading executable files (.exe, .sh, and the like). Restrict file size and file count and sanitize file names.
- Avoid unnecessary installation of CMS plugins. Install only the necessary plugins after thorough security testing.
- Install only trusted and supported CMS plugins and themes.
- Restrict access to the CMS production server only to the server administrator. Access to others should be provided strictly on the need basis.
- Disable unnecessary services and functionalities on the production server.
- Disable the CMS version number on page footer and in an error message. Also, prevent display of Web server details or CMS product details in the error log files or in exception messages. This may be exploited to launch attacks using known vulnerabilities.
- Always display a friendlier error message on the Web page without showing the details of the underlying CMS or Web server. Attacker can exploit known vulnerabilities using CMS and Web server details.
- Use strict “Access control list” (ACL) for granting permissions to secured resources, actions, functions, and systems.
- Follow the principle of “least privilege” when granting access to new users. Additional roles should be granted only on the need basis.

Security Logging and Auditing

- Log the details of all security events (such as login, logout, impersonation, secured resource access, failed login attempts). Details such as user name, time stamp, IP address, and geo-location should be logged to secured (and access-restricted) log files. This helps in tracing security incidents and identifying root causes.
- Audit all security events such as failed login attempts, login time, etc.

Content Archival and Backup

- Archive content on a regular basis by taking the backup of content snapshots from the CMS to a secure data center. Create jobs to automatically back up site content to a DR (Disaster Recovery) site. This can be used to restore the site in case of critical security breach and to ensure continuous business continuity.
- Back up the data stored in other content ecosystem components such as database, application server, Web server, and file systems on a regular basis. This helps in faster recovery in case of security incidents.

Content Classification

- Classify the enterprise content into categories such as public, private, and confidential. Public content includes content that can be viewed by all (such as Contact Us content, announcements, public policies, etc.); private content includes user-specific information (such as profile information, purchase history, etc.); and top-secret information such as trade secrets and business plans constitute confidential information.
- Based on content classification, we can apply different security policies: public content is not access restricted, role-based access controls can be provided for private content, and multi-factor access restriction can be provided for confidential content. The encryption policies, content transfer policies, storage policies, and sharing policies may also vary based on content classification.

Disaster Recovery and Business Continuity

- Establish a detailed business continuity process (BCP) to continue the business operations during an unexpected disastrous event. BCP includes disaster recovery (DR) site setup, setup of synchronization jobs, processes to handle site failover, and defining RPO (Recovery Point Objective) and RTO (Recovery Time Objective).
- Create a disaster recovery site and establish a regular synchronization process to update the DR site with latest code and configuration. The DR site can be used when the primary site is down due to a security breach.

- Update the load balancer configuration to use the DR site when the primary site fails.
- Test the DR setup to ensure that the specified RPO and RTO can be achieved.

Restricted File Permissions

- Restrict the write permission to the CMS and system files exclusively to the administrator account.
- Use restricted permissions on CMS and system folders. No file or directory should have permission level of 777 (full access to all). All CMS configuration files should be set to 444 (read-only).
- Disable auto execution of user-uploaded files and scripts.
- All user-uploaded files should be scanned for security vulnerabilities.
- All sensitive files and files in the CMS and root folders should be owned by system admin user account, and access to such folders should be restricted in Web server configuration files (such as .htaccess in Apache Web server or Web.config in MS IIS Server).

Security Monitoring

- Continuously log the security events for the production CMS system to identify any security breaches.
- Use file monitors and other audit filters to trigger notifications in case of unauthorized access.

Iterative Security Testing

- Application security is an ongoing concern. Hence, security testing should be a continuous activity.
- Various security-testing activities such as vulnerability scanning, security code review, black box testing, and penetration testing should be done periodically.
- The security test cases and vulnerability scanners should be continuously updated to keep them up to date with security best practices and able to address all new vulnerabilities.

Error Handling and Resource Handling

- The error messages should not divulge information about the Web server, application server, operating system, or the CMS system. This can be exploited by hackers to craft further attacks using known vulnerabilities.
- Convert system-generated error messages (which include error codes and error strings) to a friendly application error message.

- Instead of using generic broad exception handlers, use specific exception handlers for a given scenario. For instance, in Java we can catch specific Typed Exceptions instead of a generic catch block catching high-level Exception object.
- Release all the resources (such as file handlers, database connections, service handler) once they have been used. In Java all resource de-allocations can be done in “finally” block. If a resource has not been handled properly, it may cause a resource leak and ultimately lead to denial-of-service scenarios.

Security Governance

We need to establish a robust security governance process to address and regularly maintain the application. It should include regular patch updates, constant and real-time security monitoring, proactive vulnerability assessment, security testing, ethical hacking, and other operations.

Web Application Firewall (WAF) and Security Plugins

- Use appropriate WAF for the CMS (such as All in One WP Security and Firewall for WordPress). WAF acts as a reverse proxy and prevents many of the known vulnerabilities such as XSS, CSRF, and injection attacks.
- Use applicable security plugins for the CMS (such as the Wordfence security plugin for WordPress).

Hosted Content Systems

Hosting a CMS infrastructure over cloud can be explored wherever it is feasible. Managed cloud hosting not only manages the infrastructure but also offers a first line of defense against security attacks such as DDoS.

11.5 CASE STUDY: SECURITY TESTING FOR A CMS APPLICATION

What follows is a small case study of security vulnerabilities found in a CMS application. This will help us understand common vulnerabilities and the best-practice approaches to address the issues that may come up.

Application Background

CMS-based product support portal was developed in Java and used a JCR-compliant content repository. The portal mainly rendered product support articles and predominantly handled Web content and digital assets. For integrating with enterprise security system, a custom authentication handler was developed.

Security Testing Details

A security team carried out various tests on the product support application before the production release. The security team carried out manual code review and used vulnerability scanners and penetration-testing tools to uncover security vulnerabilities. At the end of the security testing exercise, they provided a comprehensive report of all vulnerabilities, which is detailed in next section.

Security Vulnerabilities and Remediation Measures

Here we discuss the details of security vulnerabilities found and the adopted methods to fix them. We have categorized the testing into three categories: penetration testing, manual security review, and generic security testing.

Penetration testing

In this category of testing, the security team performed the following activities:

- **Threat analysis** of the application to identify security holes and likely attack scenarios
- **Static security testing** included design-time code scanning to identify security vulnerabilities
- **Dynamic security testing** included tool-based testing of various security scenarios (such as authentication, authorization, role-based access, impersonation, etc.)
- **Application profiling:** The security team developed an overall understanding of the application and identified its main strengths and weaknesses.

As a first step, the security-testing team verified the following:

- **Threat analysis** of the application to identify security holes and likely attack scenarios
- **Static security testing** included design-time code scanning to identify security vulnerabilities
- **Dynamic security testing** included tool-based testing of various security scenarios (such as authentication, authorization, role-based access, impersonation, etc.)
- **Application profiling:** The security team developed an overall understanding of the application and identified its main strengths and weaknesses.
- Hardening of Web server, database server, application server, and CMS server. All unnecessary ports and protocols are removed and other necessary CMS hardening measures are implemented.
- Latest version of the Web server with all the necessary security fixes and patches.
- Transport-level security (SSL) was used for all sensitive operations such as authentication and user profile pages.

- Application log level changed from info mode to error/debug mode in the production environment.
- Access restricted to admin console in internal environment to administrators.
- Mod_security module is installed for an Apache Web server, which can act as a Web application firewall.
- All the necessary security patches applicable for the Web server and CMS are installed.

Detailed findings of the security testing are given below:

Default administration credentials and admin interface over the Internet

- Security vulnerability: The default administration password was not changed for the content repository. Also, the content administration interface was exposed over the Internet.
- Steps to reproduce the security incident: Access the CMS admin interface over the Internet and try the default admin password.
- Impact: An attacker who knows the product and version details can easily log in as administrator and perform malicious activities on a content repository.
- Remediation measure: It is always recommended to change all default passwords for the administrator and other user accounts. In this scenario, a new administrator account was created with a changed password. Administration port and administration URL was blocked for the Internet.

Cookie manipulation vulnerability

- Security vulnerability: The application was setting a non-sanitized user input value for a cookie
- Code Snippet: The vulnerable code snippet is:

```
String userCookieVal = request.getParameter(“username”);
.
document.cookie = "user=" + userCookieVal;
```

- Impact: An attacker can provide malicious script to launch attacks such as HTTP response splitting, site defacement, cookie manipulation, XSS, and others
- Remediation measure: User input is validated against a known white list of safe values. For instance, the white list for user name can consist of alphanumeric values.

Non-validated redirect in authentication module

- Security vulnerability: In the custom authentication module, URL parameter was used for redirecting the user

- Code Snippet: The vulnerable code snippet is:

```
String redirectValue = request.getParameter(“redirectVal”);
.
document.location = redirectValue;
```

- Impact: Non-validated redirects can trick the end-user into believing the target site. This can be exploited by an attacker to redirect the user to an attack site.
- Remediation measure: As a best practice, we should avoid redirects based on a URL parameter. If it is required, then all the redirect URL parameters should be validated against a white list. In this case, URL-based redirection was removed.

WebDAV enablement in production instance

- Security vulnerability: Web-distributed authoring and versioning (WebDAV) protocol allows the users do remote editing of content and resources. In this case, the production Web server had WebDAV enabled.
- Impact: An attacker can perform directory listing, upload/download files, and delete resources using this vulnerability.
- Remediation measure: WebDAV was disabled on the production server.

Persistent XSS vulnerability

- Security vulnerability: One of the application components stored user comments for a product article without sanitizing the input. Stored user input was rendered for all users without validation and without encoding.
- Code snippet: The vulnerable code snippet is:

```
Displayarticle.jsp
//Get user comments from URL parameter
String productComments = request.getParameter(“usercomments”);
//Store the user input without sanitization
ProductCommentsDAOImpl.persist (productId, productComments);

ProductDetailsDisplay.jsp
//Get persisted user comments
String comments = ProductCommentsDAOImpl.getComments (produc-
tId, productComments);

<!—Display persisted comments without encoding  →
<div id=”comments” class=””>
   <%= comments %>
</div>
```

- Impact: An attacker can steal user session details (such as cookies, user attributes, session ID, etc.) and the compromised session details can be used to launch sophisticated attacks such as user impersonation, session hijacking, key logging, phishing attacks, identity theft, etc.

- Remediation measure: We have already seen the remediation measures for XSS in earlier sections. For product CMS application, the following remediation measures were adopted:
    - User input validation: All user input was validated against known whitelist values. Additionally, type validation and range validation were adopted for known fields. User input was encoded before persisting.
    - Content encoding: Encoding was done at various levels: JavaScript encoding for content placed in JavaScript, XML encoding for XML content, URL encoding for HTTP header content, and HTML encoding for HTML content.
    - Secured content display: While rendering the stored content, XSS filters were used to escape/encode any reserved characters and malicious content.

HTTP response splitting vulnerability

- Security vulnerability: Non-sanitized user input value is used to set an HTTP header.
- Code snippet: The vulnerable code snippet is:

```
//Get user value for file name
String fileName = request.getParameter(“filenameval”);

//Without sanitizing, use the user values for set-
ting HTTP response header
response.setHeader("Content-Disposition","attachment;
filename=" + fileName);
```

- Impact: An attacker can include malicious code (such as carriage return, new line characters) to modify the HTTP response header and launch XSS attacks, hijack sessions, manipulate cookie values, and commit identity theft.
- Remediation measure: Sanitize the input value (file type, file size, string length, etc.) and encode the value before using it for setting HTTP response headers.

Source code disclosure vulnerability

- Security vulnerability: It was possible to browse to the source code folder and access and view the JSPs and Java code from a browser.
- Impact: An attacker can understand the internal details of application and exploit vulnerabilities.

- Remediation measure: Blocked access to the source code folder and disabled directory browsing at the Web server level. Content templates and JSPs were stored outside of the document root folder.

Information disclosure vulnerability

- Security vulnerability: Server defaults were used for error pages. Accessing a nonexisting resource produced an HTTP 404 error displaying the Apache server version, OS platform, and the port details.
- Impact: Upon knowing the system details, an attacker could exploit the known vulnerabilities for that platform.
- Remediation measure: Developed a custom error page for the application.

Code review and generic security testing

Following are the vulnerabilities found in this category:

Unsecured credential handling

- Security vulnerability: During security code reviews, it was found that credential information (such as username, password, admin URLs, application configuration) was stored in plain text within the source code.
- Impact: If the source code is compromised, attacker can access all the sensitive information.
- Remediation measure: Encrypt all the sensitive and user credential information and move it to a secured storage location.

Unbounded XML schema vulnerability

- Security vulnerability: The XML schema (XSD) does not specify a maxOccurs value for the element.
- Code Snippet: XSD contains the following entry:

```
<xs:element name="section" maxOccurs="unbounded"/>
```

- Impact: This vulnerability may potentially be exploited for creating resource exhaustion and denial-of-service attacks.
- Remediation measure: Specify a maximum upper bound value for all the elements.

XPath injection vulnerability

- Security vulnerability: Few user credential details were stored in the XML file and they were used in custom authenticator for login.

- Impact: Attacker can use XML special characters (such as apostrophe (‘), less than sign (<), greater than sign (>)) to inject malicious code similar to an SQL injection.
- Remediation measure: Remove all sensitive information from the XML file and validate the input (for example, if there are XML special characters, escape it) before appending it to xpath expression.

Application Retesting

Once the remediation measures were applied, the application was retested with the same test data to ensure that all security vulnerabilities were indeed addressed.

11.6 CHAPTER SUMMARY

- The chapter discussed security aspects of content systems.
- Cross-site scripting (XSS) attacks may execute malicious scripts on client systems. XSS can be persistent or non-persistent.
- For preventing XSS we can use the following techniques: design content security policy for resources, leverage OWASP AntiSamy library, use application filters, perform input validation.
- SQL injection attacks use crafted SQL strings to perform database operations unknown to the end-user.
- Effective ways to mitigate SQL injection attacks are by validating user input, using prepared statements, using access restrictions on application schema, and filtering malicious URL parameters in Web server configuration files.
- DoS and DDoS attacks overload system resources through request flooding.
- In order to prevent DoS and DDoS attacks, we can use network filters, cloud hosting, and CMS-level preventive measures.
- CSRF exploits the trust between the Web site and the user agent. An attacker can extract sensitive information through embedded commands.
- In order to prevent CSRF, we need to do application-level validation and use request authorization tokens.
- To prevent clickjacking, we can set X-FRAME-OPTIONS to SAMEORIGIN.
- CMS platforms use built-in authentication or are integrated with LDAP for authentication.
- SSO can be achieved through centralized security managers and SSO tokens.
- Permission model specifies the fine-grained access control for various content resources.
- Resources, in the context of content management, are essentially content objects (such as templates, workflow, jobs, pages, etc.) and content functions (such as authoring, publishing, administering, etc.).

- For defining a robust permission model, we need to define the permissions, map permissions to roles, and then map roles to users.
- Users inherit permission through their assigned roles.
- The key security best practices are multi-layer security, robust account management, proactive vulnerability scanning, security testing, security code reviews, CMS patching, CMS hardening, security logging, content archival, DR setup and BCS definition, restricting file permissions, error handling, security governance, Web application firewall, and hosted deployment

Chapter 12

Content Infrastructure and Performance Optimization

The infrastructure of the content ecosystem plays an important role in the availability, scalability, and performance of the system. The infrastructure includes the hardware specifications of all environments. Sizing and capacity planning should be a well-planned activity to be performed during the architecture-planning phase of the project. We have seen many instances wherein the system was not able to scale to high-volume content and user traffic, leading to program failures.

The performance of the Web site has a huge impact on end-user experience. Overall success of the content strategy very much depends on its performance perceived by the end-user. Content performance would also impact its distribution channels (such as services, feeds). Content usability and effectiveness, the key success criteria for content strategy, may also be affected due to content performance. Search engines also consider site performance in their rankings. For content-driven Web sites, the performance of content directly impacts the overall site performance.

In the first part of the chapter we discuss various elements of infrastructure architecture such as sizing, deployment architecture, and disaster recovery setup. In the second part of the chapter we discuss various aspects of content performance. We look at various performance optimization techniques such as optimal content design, optimal workflow design, caching, database-level optimizations, content caching, monitoring and notification, infrastructure-level performance optimization, etc. The chapter concludes with a look at content performance KPIs and performance testing.

Infrastructure architects, content architects, enterprise architects, performance engineers, and CMS developers will find this chapter useful.

12.1 CMS INFRASTRUCTURE ARCHITECTURE

In this section we discuss the infrastructure elements of CMS. Infrastructure architecture mainly consists of deployment architecture and disaster recovery setup.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

Deployment architecture depicts the infrastructure used for solution deployments. Though deployment architecture normally covers the hardware setup and configuration, this helps us understand the CMS solution deployment ecosystem.

Infrastructure Sizing

CMS system sizing covers the servers in both authoring and publishing environments. Without properly sized hardware, CMS platforms are not able to meet its intended goals and SLAs.

Sizing Questions

Sizing activity begins with an understanding of the expected/anticipated load. Given below are the key questions that help us understand the load. We have categorized questions separately for the authoring instance and the publishing instance. Publishing live instance is also referred to as delivery system, serving content to Internet end-users.

Sizing questions for authoring instance

- What is the maximum content volume created in the authoring instance?
- What is the maximum number of authors who use the authoring instance?
- What is the maximum concurrent author logins to the authoring instance?
- What is the frequency of content publishing?
- What is the volume of assets and documents? What is the average number of asset renditions needed for each asset?
- What is the maximum number of templates along with their complexity and workflows needed by the solution?
- What are the integration requirements for the authoring instance?
- What are the requirements for multi-site deployment and remote deployment?
- What are the various pre-production environments (such as development, SIT, QA, UAT etc.)?
- What are the maximum concurrent transactions?
- What will be amount of custom coding (custom extensions, custom libraries, custom templates, custom components, etc.)?
- What is the anticipated content volume growth?
- Is there a requirement for the authoring instance to be accessed outside the intranet?

Sizing questions for the publishing instance

- What is the peak user load and expected user growth rate?
- What will be the maximum number of concurrent users?

- What are the performance, scalability, support, and availability SLAs?
- What are the various access channels for the application?
- What are the various geographies served by the application? And what are the SLAs in those geographies?
- What is the maximum number of page views per hour?
- What are content search requirements?
- What are the performance SLAs?
- What are the caching requirements?
- What are content retention and archival related regulation policies?
- What is the expected CPU and memory utilization during peak hours?

Sizing Exercise

Once we identify the workload and anticipated growth from above questions, we will size the CPU, memory, disk storage, network bandwidth, and database for the authoring and publishing instances. Application complexity will also be considered along with load values for sizing the servers. Most CMS vendors provide benchmark numbers that can be used to match with the obtained load requirements to arrive at the appropriate sizing values. The following are the key considerations in CMS sizing:

- Publishing servers must always be clustered to support failover and scalability.
- Adopt multi-layer caching to cache content and content fragment at all layers to improve performance and scalability.
- If the site is asset intensive, consider using content delivery network (CDN) that provides efficient asset caching across geographies.
- Evaluate cloud hosting along with on-premise hosting to find the most appropriate hosting option.
- Set up a robust monitoring infrastructure to monitor the health of internal servers and to monitor the production live instance for performance and availability SLAs.
- Use the optimal server configuration recommended by the CMS vendor such as connection pool settings, thread settings, cache settings, session replication settings, etc.

Basic Concepts of CMS Deployment Architecture

Normally content management systems will have content-authoring and publishing servers (sometimes referred to as content delivery servers). Authors, writers, and reviewers use the content-authoring server for content creation and review. CMS developers can create CMS components and page layouts using the content templates on the authoring server. Administrators will size the authoring servers appropriately to scale for appropriate content volume and user volume. Usually content-authoring servers will be within the enterprise's internal network, protected by an internal firewall and accessible only to intranet users. If external parties (such as creative agencies, external authors, and external reviewers) need to access the authoring server, a secured authoring server interface will be exposed over the Internet. Once content is reviewed and approved by all concerned stakeholders, final content will be published to the publishing server through the publishing workflow. During publishing, content also undergoes the required transformation to an appropriate format (such as HTML, JSON, XML, etc.). Publishing or delivery server is mainly for content delivery. Content delivery servers are normally present in a demilitarized zone (DMZ) so that they can be accessed by Internet users. They render the reviewed and approved content for end-users. Content delivery servers will also use Web servers for optimal performance (by using static page and asset-caching features of the Web server). These servers will be sized to support the end-user load and the peak load. For simple scenarios (such as intranet sites with minimal content volume), we can host both authoring and publishing servers in the same server machine. In the vast majority of cases, we will deploy authoring and delivery servers on separate servers (each with a clustered mode) so that the publishing server is accessible by Internet users and the authoring environment is protected by a corporate firewall. This helps in scalability of individual instances with optimal performance; the setup is able to support a high number of external users (viewing the content on the publishing server) and a large number of authors (authoring content on the authoring server). Each of the server instances can be clustered and sized based on their access load. Separating authoring and publishing instances will also help in maintaining robust security. We can enforce stricter and role-based granular security access for each of the content functions (such as creation, authoring, review, and publishing) on the authoring server. We will not be exposing the content administration functions over the Internet, as the authoring instance will be available within an internal firewall. On a publishing server we can restrict the resource permissions and enable access to only specific ports and protocols. It would also be easy to perform administration activities on authoring and publishing servers when they are separate.

CMS Deployment Setup

For optimum scalability and availability, we use a clustered deployment model for both authoring and publishing instances. Based on the user and content load, we will size the servers and use multiple server nodes as part of the cluster. A multi-node cluster will also provide redundant hardware to handle any unexpected server failure. The load balancer distributes the requests based on server availability and for optimal performance.

A sample CMS deployment setup is depicted in Figure 12.1.

In the deployment setup shown in Figure 12.1, we have clustered the authoring servers in a secured internal zone. Content authors, reviewers, and approvers use these authoring servers to author, review, and approve content. Once content is approved, it is published to the clustered publishing servers hosted in the DMZ. Publishing

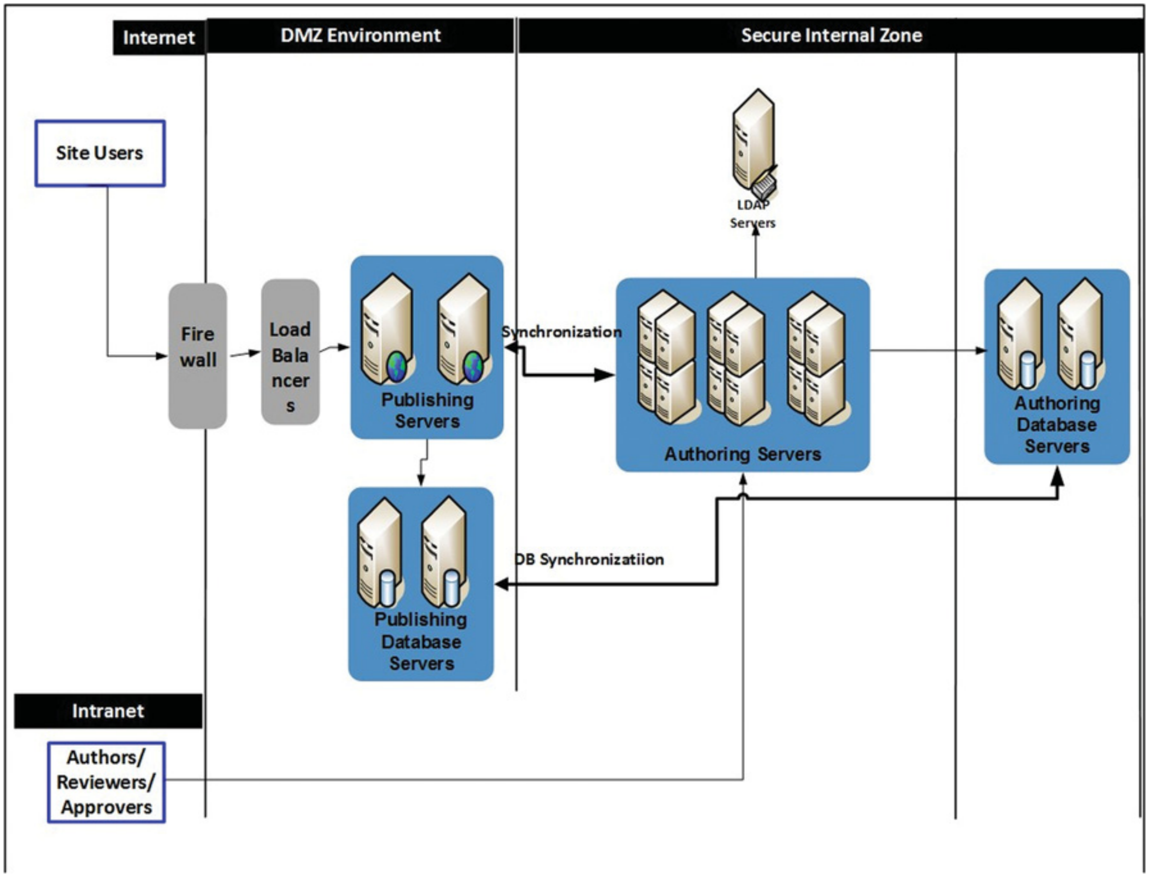


Figure 12.1 CMS Deployment Setup

database servers are also hosted in the DMZ. Changes happening directly with the publishing database servers are synchronized with the authoring database servers on a regular basis. Load balancers evenly distribute the user requests to the publishing server.

In order to ensure high availability and appropriate scalability, the content infrastructure should be planned based on user load, content volume, page requests per second, etc.

Deployment best practices

- We need to have following environments for efficient deployment and release planning:
  - Development environment for developers to develop and locally test the CMS components and libraries and check in the code to the source control system
  - Integration environments for integrating code from all developers and performing integration testing
  - QA environment for end-to-end system validation
  - Authoring environment for authors to create content
  - Publishing environment consisting of preview and live instance; the preview will be the staging environment for final validation, and live environment for the delivery to public users
- Set up a DR environment that mirrors the production live environment in terms of code, content, and configuration; set up regular synch-up jobs to synchronize content between production site and DR site
- Both author instance and publish instance should be appropriately scaled

Disaster Recovery Setup

Disaster recovery setup involves the creation of a mirror site in a remote (preferably distant geographic) data center. A disaster recovery (DR) environment, which is an exact replica of the production systems, would serve as failover environment. The DR site will be used to handle any unexpected disasters occurring at the primary site, and it is an important element of business continuity planning process (along with backup and recovery processes).

We can either design a custom approach or leverage the product's replication feature to ensure that content is synced between the primary site and the DR site. Content sync can either happen on demand or with a fixed frequency between the production and DR environments.

Other options for content synchronization between the main site and the DR site are:

- File-based copy of the content store between production and DR environments. This will copy all the pages, templates, assets, content, assets, configurations, and related files and documents.
- Synchronization between the content repositories can be done using tools.

All these options require that code and configuration updates happen simultaneously between production and failover environments.

During initial DR setup, we will transfer complete copies of files and configurations from the primary site to the DR site. Subsequently we will only do partial copying (of the updates).

12.2 CONTENT PERFORMANCE OPTIMIZATION

In this section we will explore various content performance optimization design and best practices.

Optimal Content Design

- It is always recommended to create modular content chunks instead of a single monolithic chunk. This design not only enhances reusability but also helps in performance optimization across various channels. As chunks can be easily cached and refreshed, it improves page performance.
- Design and tag content with semantic metadata. This helps us bring the most relevant content for a given context. Faster content discovery improves the performance of the content search process.
- Use adaptive content design techniques to deliver optimized content and asset variants for a given context. This improves the content performance across various channels.
- On the presentation side, use responsive Web design (RWD) techniques for optimal performance on all channels. Responsive Web design will optimize the page layout for the corresponding device, and adaptive content will render the most optimized version of content for a given context.

Optimal Page Design

- The page design should adopt the *lightweight* and *lean* design philosophies. Key site pages such as gateway pages, home pages, and landing pages (referred to as entry pages in the remaining sections) should not be cluttered with too many site sections. “Heavy” pages impact the page loading times.
- Instead of stuffing too much content and functionality on the entry pages, we can provide a robust navigation model and information architecture to help users discover content easily and quickly. This includes contextual left navigation links, breadcrumbs, expanded footer (containing important links), relevant context menus, quick links, personalized links, saved bookmarks, etc. This will not only eliminate unnecessary content on the entry pages but also bring more contextually relevant and personalized content to the user.

- Use Web analytics to continuously monitor the usage of content on the entry pages and remove less-used content and move it to sublevel pages.
- For content page performance optimization, use the following techniques:
  - Minimize the number of JS and CSS files by merging all JS files into a single master JS file and all CSS files into a single master CSS file.
  - Minimize the master JS and CSS files to reduce their size. We can use tools such as YUI compressor, JSMin, and Speedy for minimizing JS and CSS files.
  - Place the master CSS file at the top of the content page and the master JS file at the bottom of the content page to optimize perceived performance.
  - Many CMS platforms provide built-in support for these optimizations, and we can leverage them (for instance, Drupal CMS provides "Aggregate and compress CSS files," "Compress cached pages," and "Aggregate JavaScript files" configurations for this). Open source Apache mod_pagespeed module can be leveraged to transparently re-write web pages and use performance optimization filters.
- Optimize page assets:
  - Always use the most optimal image format and correctly sized image for a given rendition. A mobile page will require a smaller-sized image as compared to the regular Web page. Hence, the asset publishing workflow should be designed to use images of various resolutions based on the target delivery platforms. The presentation and transformation engines should conditionally select the most optimal image format and image rendition based on the target platform.
  - Adaptive image technique can render the image in the correct size automatically based on the target device's dimensions. We discussed one such technique as part of the adaptive content discussion.
  - Lazy-load images for improved performance. Some CMS platforms provide built-in support for this lazy-loading (such as the Image lazy loader module in Drupal CMS).
  - Compress images to reduce their overall size. A PNG format is one of the most optimal image formats.
  - Use CSS sprites to reduce the number of image requests for a given page.
  - Cache the assets at the content delivery network (CDN) layer, which will reduce the load on the origin server and optimize asset delivery across geographies.
- All duplicate content should be removed and all broken links should be fixed. We can use the link validator before publishing content to ensure that a content page has valid links.
- When the page is displaying large amount of records or results, it is always recommended to use a paginated result display. This limits the size of the initial result set and improves initial page performance. Search result pages and content list pages can adopt this technique.

- Wherever possible, content sections should load the data on demand in an asynchronous way. This provides a non-blocking page load.
- Leverage the Web server's compression feature to compress the Web page before it is rendered to the end-user.

Optimized Publishing Workflows

- Full site publishing should only be used during initial site creation. All subsequent publishing workflows should use partial site publishing to only publish the updated or new content. This approach will optimize the publishing process.
- Publishing workflows should also publish the most optimal content format based on the target delivery platform. For instance, a Web platform needs HTML, whereas the file system need a PDF format.

Database-Level Performance Optimizations

- Create and test the database indexes for optimal database performance.
- Profile the queries fired by CMS and test its performance at high data volume. Optimize the query performance.
- Identify most frequently executed queries for popular site pages and optimize them.
- It is recommended to create a separate schema for application-level data.

Content Caching Design

- A multi-layer caching should be designed at all applications layers. Content should be cached at CDN layer, Web server layer, service layer, database layer, and CMS layer.
- Content should be cached at the chunk/fragment level instead of the page level. With this approach we can selectively refresh the cache for optimal page performance. Content cache should be invalidated when the cached content is changed.
- Cache the page assets based on their update frequency. Images and scripts that do not change frequently can be cached for extended duration. We can cache them at the Web server layer and the CDN layer.

Monitoring and Notification Setup

Performance is not a one-time activity. We need to continuously monitor the content performance for early identification of any performance issues. We could leverage any of the existing monitoring products available in the market for application performance monitoring (using tools such as CA Wily Introscope, Dynatrace), infrastructure performance monitoring (using tools such as Ganglia, Splunk), and live site performance (using tools such as Gomez). Web analytic frameworks can also be used for page performance reporting. We should configure the performance SLA, performance thresholds, and the notification feature as part of the monitoring setup. Key points in performance monitoring setup are as follows:

- As part of the infrastructure setup, we need to set up a robust internal server monitoring and notification infrastructure that will check critical server health check parameters (such as CPU and memory utilization, network bandwidth consumed, throughput, request execution time, cache hits/misses, etc.). All performance incidents need to be reported to the site administrator in real time so that they can be addressed quickly.
- We can also monitor the performance of content APIs and database queries in each layer.
- For globally distributed applications, we should also set up a real-time external monitoring infrastructure that constantly monitors the production applications and production pages and notifies in case of any SLA violations.

A robust monitoring framework will provide early detection of performance SLA violations and alert site administrators to take quick corrective actions.

Proactive Identification of Memory Leaks

- Use code-profiling tools and profile the code with average and peak loads for extended duration of time to discover memory leaks. Tools such as VisualVM can be used for code profiling and for monitoring JVM metrics.

CMS-Level Performance Optimizations

- Disable all unnecessary background jobs and services that are not required for the application.
- Disable unnecessary CMS modules and components that are not needed by the application. For instance, we can disable modules such as Statistics and Update Manager in the production instance on Drupal.
- All the batch jobs, backup jobs, archival jobs, and synchronization jobs that synchronize content to disaster recovery sites should be run during the time of minimal site traffic.
- Ensure that CMS settings (such as pool size, caching settings) follow the vendor-recommended best practices.

Logging

Ensure minimal logging (setting log levels to only error) in the production environment.

CMS Caching

Leverage CMS built-in caching features:

- Content and content fragment caching for frequently used pages
- Caching the results of frequently used content queries
- Caching the content retrieved from content service calls
- CMS also allows the configuration of cache expiration time and auto cache invalidation upon updates to dependent elements
- Caching of data retrieved from CMS database.

Search-Engine-Level Performance Optimization

- Search engine spiders crawl and index CMS content. This could add additional performance overhead for the site and CMS system. Hence we need to fine-tune the crawling frequency. Configure the search crawlers to crawl CMS content during least traffic.
- Search engines also allow configurations to crawl only the updated sections instead of full crawl. These configurations can be used for efficient crawling.
- CMS can notify the search engine when new content is published to facilitate on-demand crawling.

Infrastructure-Level Performance Optimization

- Do proper hardware sizing and capacity planning based on the expected user loads. CMS vendors may also specify the benchmarking numbers, which can be used as reference. The Web servers, application servers, database servers, CMS hardware, and network should be appropriately sized and tested for normal load and peak loads. It is always recommended to use clustered configuration for authoring and publishing servers to ensure appropriate workload distribution.
- A reverse proxy can be used in front of the CMS Web server, which can be used for serving cached pages and cached assets. This not only reduces the load on CMS Web server but also serves content and assets faster.
- We can enable the compression at the Web server level, which improves the performance of the content rendition, especially in low-bandwidth regions. For instance, Microsoft IIS Web server supports static content compression configuration, which can be used for this.

- Leverage the cache offered by the CMS to cache the content and content fragments.
- Use content delivery network (CDN) for forward-caching the static content and assets. CDNs may provide optimal performance through its geographically distributed nodes and hence would be most effective for achieving good performance across multiple geographies.
- Configuring session and cache replication across cluster nodes for optimal performance.
- Configuring optimal load balancing strategy.
- Fine-tune application server settings such as connection pool settings, JVM heap size, thread settings, and other vendor-recommended parameters.
- Providing vertical and horizontal scaling using cloud-based infrastructure.
- Set appropriate cache headers for the static digital assets such as images, videos, and documents so that user agents (browsers) can cache the digital assets for optimal performance. For instance, in the Apache Web server, the following configuration in a .htaccess (or in httpd.conf) configuration would set the cache headers for digital assets for one week and JS/CSS files for two hours.

```
# 1 Week cache
<FilesMatch"\.(jpg|jpeg|png|gif|swf|ico|pdf|avi|mp3|wav|wmv)$”>
Header set Cache-Control "max-age=604800, public"
</FilesMatch>
# 2 Hours cache
<FilesMatch "\.(js|css)$”>
Header set Cache-Control "max-age=172800"
</FilesMatch>
```

- We can also instruct the Web servers to compress the output and set the expiration for the digital assets using following headers in the Apache .htaccess configuration file.

```
# Enable compression
<IfModule mod_deflate.c>
 AddOutputFilterByType DEFLATE text/html text/plain text/css
application/json
 AddOutputFilterByType DEFLATE application/javascript
 AddOutputFilterByType DEFLATE text/xml application/
xml text/x-component
 AddOutputFilterByType DEFLATE application/xhtml+xml
application/rss+xml application/atom+xml
</IfModule>
```

```
# Add expiration period
<IfModule mod_expires.c>
ExpiresByType image/gif                "access plus 1 week”
 ExpiresByType image/png                "access plus 1 week"
 ExpiresByType image/jpeg               "access plus 1 week"
 ExpiresByType video/mp4                "access plus 1 month"

 # CSS and JavaScript
 ExpiresByType text/css                 "access plus 1 week”
 ExpiresByType application/javascript   "access plus 1 week”
</IfModule>
```

12.3 CONTENT PERFORMANCE KEY PERFORMANCE INDICATORS (KPIS)

Continuous real-time tracking of content metrics provides vital insights about the content usage and its performance. We need to identify all the content performance KPIs during content the strategy phase and design the mechanism to track and report those metrics. In this section we look at the content performance KPIs. This will help the content strategists and content administrators define and implement the reporting mechanism for these KPIs.

Collecting Content Performance KPIs

Tagging content and assets is one of the best ways to collect the content performance metrics. We can tag entire page content or we can add Web analytics tags at the content chunk level. For instance, we may track the metrics such as number of clicks, number of downloads for buttons and click-to-action items, sharing metrics for blogs and articles, and the like. Web analytics scripts are used for tagging content and digital assets.

Another way to collect the content performance metrics is to use real-time performance-monitoring tools such as Gomez. These tools will hit the content pages with same context as the end-user (geography, user agent, network, etc.) to accurately measure the performance perceived by the end-user.

Content Performance KPIs

Table 12.1 shows the performance KPIs that we can use for various content types and asset types.

Note: The term “performance” is used in a broad sense while defining these KPIs. It not only includes run-time response times but also the performance of the overall CMS system, usability, and effectiveness of content.

Table 12.1 Content performance KPIs

Content Type	Performance KPI
User-generated content (blogs, articles, wiki, forum posts)	• Number of shares • Number of likes • Number of incoming referred links • Number of views • Time spent on each of the articles, blog posts • Number of comments • Overall rating for the content
Click-to-action items (submit buttons, images)	• Number of clicks • Number of downloads • Number of conversions
Home page content	• Overall page load time • Time to first byte • Number of server requests • Overall server response time • Asset load time • Number of assets needed by the page • Perceived page load time • Exit rate/bounce rate
Content workflows (publishing workflow, authoring workflow, translation workflow)	• Total number of steps in the workflow • Number of manual and automated steps • Total time taken for completion of the workflow
Digital assets (image, document, video)	• Asset download time • Asset load time • Time spent in viewing (for videos) • Number of downloads

12.4 CONTENT PERFORMANCE VALIDATION

We can use regular performance-testing types such as average load testing, peak load testing, stress testing against the CMS systems, and the content pages to test its performance. Open-source tools such as Apache JMeter can be used for load testing. Tools such as Google PageSpeed Insights and Yahoo YSlow will analyze the page from the performance standpoint and provide improvement suggestions.

Another effective way to test the content effectiveness is to adopt multivariate testing and A/B testing in which multiple variants of content, buttons, and images are tested. This will help in determining the most optimal and effective layout, positioning, and usability of content and assets. Omni-channel content should be tested on all supported browsers and devices.

Load metrics

We need to predict the user load and traffic (for new applications) or collect the load metrics (for existing applications) so that we can model and test various performance scenarios. In this regard, what follows are some of the load metrics required for performance validation:

- Expected page response time: This helps in performance validation and sizing of servers
- Logins per day: This helps in designing scalability values and scalability testing
- Active site usage period: This helps in designing session configurations
- Average time per Session: We can design session values (such as session timeout, session idle timeout, etc.) based on this
- Average page views per session: We can design optimal cache size and scalability values based on this
- Concurrent users (concurrent user = arrival rate x average session time): We can design optimal cache size and server hardware size based on this
- Total page views per second: This helps in sizing the memory, CPU, cache, and storage for the servers
- Average transactions per page view: This helps in sizing the memory, CPU, cache, and storage for the servers

12.5 CONTENT-RELATED BEST PRACTICES

In this section we take a look at some of the proven CMS and content best practices adopted in real-world programs. This can be used as a checklist for validating content and CMS design in large-scale programs. Best practices provided in this chapter can be used along with content checklist in Appendix B. We begin with a look at various content best practices for both core content and CMS. In the next section we look at search-related best practices from various dimensions such as search user experience, personalized search, content search monitoring, and search performance.

Content Best Practices

- **Separation of content and presentation:** Core content should be separated from the presentation logic. During content creation, created content should be stored in an open standard such as XML or JSON. Presentation templates and transformation engines should add the presentation logic (such as styles, alignment, formatting) as per the requirements of the target device or platform. This makes content reusable across various presentation devices.

- **Optimized digital assets**: Develop workflows to resize the image to appropriate resolution and size for optimal rendering on a target device. Many CMS and DAM systems provide built-in workflows to provide various resolutions of the image.
  - Minimize embedded text within graphics. This increases reusability in localization scenarios.
  - Use graphics, icons, and text, which is generally accepted in all cultures and geographies.
  - Use relative paths for images and HTML content.
  - Use workflows to create multiple renditions of the image varying in size, resolution, and dimension. We can use these renditions based on target device and form factor.
  - The frequency of updates for digital assets is relatively low compared to the same for content. This provides opportunity to cache the assets at various layers such as Web server, CDN, and browser for optimal page performance.
- **Content links**: Use descriptive titles for the content links to enhance readability and SEO. It should provide keyword-based brief description for target content instead of generic label such as "link."
- **Standards-based implementation**: Authoring and publishing processes should follow open and industry standards so that we can easily extend, integrate, and manage content. Publishing standards (such XML, JSON, HTML), open standards in integration (such as SOAP, REST, JSON), Web standards (such as HTML, W3C), and accessibility standards (such as WCAG 2.0) should be followed during the content lifecycle.
- **Continuous content improvement**: Monitor content in its entire lifecycle and address gaps related to outdated, duplicate, or incomplete content.
- **Content development:**
  - Use iterative development and continuous testing of components. We can discover the issues early by adopting an iterative testing approach.
  - Use appropriate loggers in the components to ease troubleshooting process.
  - Use the CMS-provided extensions to extend the core functionality (such as security extensions, integration extensions) instead of creating custom functionality.

Content Best Practices Checklist

In this section we look at tenets of good enterprise content, which form the key ingredients of content strategy. Content authors, designers, reviewers, and approvers can use this as a checklist while creating, designing, reviewing, or testing content.

The main characteristics of good content are:

- **Usability**: The primary goal of content is that it should be usable. Content should be able to convey the required information, help users make informed decisions, or influence customer behavior through call-to-action components. Keep the content messaging simple. Content should be rendered on all devices and should be easily readable and understandable. Design, structure, labeling, accessibility, branding, findability, relevancy, and consistency are among the important factors that affect content usability.
- **Customer-centricity**: Content should cater to the needs and wants of a specific user persona. Content should also be personalized based on user preferences.
- **Simplicity and clarity**: Content should be easy to understand for a common audience. It should be relevant in a given context.
- **Messaging strategy**: Content messaging strategy should communicate broad information, voice, and tone creatively, consistently, and persuasively to the user. The message can convey main intent (for example, we could say that "we have lowest priced rental cars") or messages can also provide unique value proposition (the message "we have largest fleet of rental cars than any of our competitors in this region" differentiates from the competition). Messages should resonate the established voice and tone for the organization.
- **Conciseness**: The verbiage should be crisp and easy to grasp. For lengthy content, it is better to break down it into various sections and subsections with appropriate labels for better readability.
- **Findability and searchability**: Users should be able to reach relevant content quickly in minimal number of steps or clicks. This can be achieved through appropriate metadata tagging and optimized navigation model and user-friendly information architecture. Ensure that content and documents are easily findable. This can be achieved through number of techniques such as enhanced content labeling (using meaningful headline/title tags, page metadata) and context-sensitive menus. The primary tool to enhance information discovery is the enterprise search. By improving the search effectiveness, making search fast and easy, and positioning the search as the main information discovery tool, it is possible to make the search as the primary gateway of information discovery. Search can be enhanced to include search aids such as auto-completeness, synonym support, guided navigation, faceted search, spell-corrections, etc. Intuitive navigations also aid content findability. Descriptive menus, friendlier left navigation, and breadcrumbs will help users find the relevant content faster. We can also facilitate information discovery through quick links content, popular/trending topics, frequent downloads content, etc. Search engines also leverage the metadata to serve relevant content.
- **Contextual and relevance**: Content should be relevant for specific user persona and should be aligned with the context. For instance, a product support person should see "Supporting Material" in product pages showing product support articles.

- **Consistency**: The voice, tone, message, color scheme, structure, and format of content should be consistent throughout the delivery channels. Visual style guides will bring in uniformity in content, and reusable content templates and layouts will provide uniform structure for the content. Content labeling and categorization should also be consistent throughout the online channel. Authoring aids such as copy deck would also help authors create content in a desired way. Some of the metadata values (such as product names, country names, languages, etc.) and content values need a stricter governance to bring consistency. This can be achieved through centralized repository for controlled vocabulary and taxonomy.
- **Accessibility and Omni-channel enablement**: Content should be able to cater to multiple channels either automatically or through configurations. Content should also be accessible on all supported browsers and devices.
- **SEO friendliness:** Content should incorporate SEO best practices to include friendlier URLs, page metadata, and titles to enhance discoverability by external search engines. It is essential to expose site taxonomy to enhance content visibility. Ensure that content does not have any broken links and provide descriptions and keywords for all content and digital assets.
- **Content readability**: Content description should be easy to understand. For complex and lengthy content, create a bulleted summary for quick understanding. Use the right tone and language to enhance readability.
- **Content messaging and communication**: Content should use infographics, visualizations, and rich media assets (such as videos, marquee banners, image accordion, etc.) to improve user understanding and to communicate the message more effectively. For sequential content, a bulleted-list summary or a numbered list can be used. A glossary of abbreviations and acronyms can be provided to prevent any ambiguity/confusion in communication.
- **User engagement and user adoption**: Content should be easily connectable with end-users actively through blogs, feeds, and leverage CMS platform to build wiki to harness collective intelligence (through co-creation). Content platforms should be integrated with social channels to engage users on their platform of preference. Content should drive user adoption.
- **User information content**: This includes content that provides the right information to users. FAQs content, step-by-step how-to articles, video tutorials, product support, product compatibility matrix content, security policy content, privacy information, and similar content fall into this category.

Content best practices

The key content best practices are:

- **Content chunk strategy**: It is recommended to create modular content sections (using content chunks) with meaningful labels instead of creating monolithic, lengthy content. This improves readability and makes content reusable across multiple channels. If there is large monolithic content, it should be broken down into logical sections with meaningful heading labels tagged with appropriate metadata. This further enhances content reusability.

- **Deliver personalized and contextualized content**: Content can be automatically personalized based on user attributes or can use explicitly specified user preferences.
- **Create user-centered content**: Content should be easily usable for the intended audience.
- **Optimize content ecosystem**: This includes refining metadata strategy, information architecture, and workflow strategy for optimal performance of content.
- **Adopt content reusability strategy**: Design content so that it can be reused. This will be discussed in detail in the upcoming sections.
- **Identify and eliminate duplicate, unnecessary, and outdated content:** Using proper metadata and tagging, we can identify and eliminate duplicate content.
- **Intuitive information architecture and information discovery**: A robust information architecture strategy includes friendlier content grouping and intuitive content organization. It also encompasses intuitive content categorization, navigation aids, personalized navigation, expanded header/footer, user-centric site map, and others. Complex and lengthy content can be broken down into easily readable content fragments. The primary aim of information architecture is to minimize the steps/clicks needed to find the right information, which can be achieved through simplified workflows. Provide aids and tools to influence users' decisions.
- **Content tracking and optimization**: Content analytics should be used to regularly monitor the analytics reports to understand the usage and challenges with existing content and use it to optimize the content strategy. Additionally, it is essential to leverage user feedback, multivariate testing to evaluate the usefulness of content, and use this feedback to improve content.
- **Content presentation**: Content should be presented with a uniform and consistent visual layout and design.

CMS Best Practices

- **Robust requirements elaboration**: CMS interacts with various other systems such as security systems, translation systems, services, metadata systems, etc. Hence, it is important to gather the complete set of requirements from all aspects. Once the requirements are elaborated, it should be reviewed and signed off on by all concerned stakeholders. All nonfunctional requirements such as CMS performance, scalability, and availability should be clearly defined along with their SLAs.
- **Business-friendly processes**: The business-critical CMS functionality such as CMS administration, workflow configuration, campaign management, and

marketing functionality should be made business friendly. This includes features such as configurable user interfaces, easy-to-use business processes, etc.

- **Standards-based platform**: CMS should adopt standards-based development and integration practices. This includes JCR-compliant content repository and access APIs, consuming and exposing services through REST and other standards.
- **Open architecture**: The CMS design should be based on open standards such as DITA, XML, and JSON that would create a standards-based content platform.
- **Minimization of content silos**: Create a unified view of all enterprise content to enhance reusability and ease content management. This also avoids the content duplication. Wherever possible, migrate content from all sources to a centralized content repository.
- **Centralized content access**: If an organization has multiple content systems, it is essential to consolidate them so that we can provide uniform content classification and single point of access to the users. We can also easily manage and control the access to the centralized system.
- **Success metrics**: We should define the key success factors to track the effectiveness of the CMS. It could be something like improvement in content publishing time, improvement in content approval times, improvement in process execution time, and the like. These metrics should be able to quantify the effectiveness of the CMS, and business stakeholders should be able to track the effectiveness of using CMS platform for their content needs and the business goals.
- **Content governance**: There should be a robust content governance structure with well-defined role-responsibility matrix. This ensures that various stakeholders and users are aware of the ownership and responsibilities in the content management lifecycle. This also helps in establishing a standard operating procedure (SOP) for responding to emergency situations.
- **Metadata-driven content discovery**: Metadata tagged with content should be used for accessing content. Metadata hierarchy should be defined such that it can add meaning to content.
- **Reusability**: During CMS design, we need to focus on modularity and reusability of the underlying components. For instance, creating flexible content templates can be reused for creating multiple content structures.
- **Content sessions**: For optimal memory and performance, session idle timeout value should be configured.
- **User interface:** An administrator should be able to configure the UI elements and CMS should be able to personalize the UI features based on user attributes.
- **Content services**: CMS should support SOAP- and REST-based services for exposing and consuming content services.

- **Security**: CMS should be able to provide built-in features to prevent security attacks such as cross-site scripting (XSS), click jacking, CSRF, etc.
- **Content-authoring template**
  - Minimal number of templates should be created to cater to a wide set of site pages.
  - Templates should be flexible enough to be reused across a wide variety of scenarios.
  - Templates should be able to support open standards such as XML, DITA, and JSON for storing and publishing content.
  - Content authors should be able to easily extend and reuse the templates for creating future content needs. Flexibility can be added through usage of repeatable sections, optional elements, flexible elements, and similar features.
  - While designing templates, group the semantically related elements into one logical section. This will enhance the usability of the template and speed up the content creation process. For instance, image path, image metadata, and image description can be logically grouped together because they form the key components for a marquee section.
  - Place the frequently updated fields at the top of the template so that authors can easily locate the fields and update quickly.
  - Provide help for each of the template fields explaining the usage scenarios and restrictions (such as maximum length, allowed values, etc.) so that authors can refer to the help while authoring.
  - When content has to be published as HTML, we can provide rich text editor (RTE) for authoring rich content in authoring templates. The RTE can be integrated with the specified visual style guides to ensure visual consistency.
- **Content model**
  - The content and asset model should be flexible to support all required content structures for the page.
- **Content caching**
  - Adopt layer-wise caching for optimal performance.
  - We should not cache secured content (such as user profile page), which requires security checks.
- **Workflows**
  - Ensure that all key business processes related to content are modeled as content workflows.
  - The processes that the editorial team (authors, reviewers, marketing) uses on a daily basis should be given high focus. We need to ensure that all these frequently used processes are optimized as they are used on a frequent basis.
  - While modeling the business processes as content workflows, we need to explore all opportunities to optimize the process steps. Some of the effective optimizations are:
    - Auto-approval of some of the trivial activities
    - Time-based auto forwarding of process steps

    - Auto-routing of an approval step if the approver is not available
    - Parallel execution of independent workflow steps
    - Asynchronous invocation of long-running services with ability to pause and restart the workflow (such as during content translation)
    - Workflow should be able to handle all possible contingencies using fallback mechanisms.

- **Release management**
  - In order to minimize the risk and to ensure faster time to market, it is recommended to perform content releases in iterations.

12.6 CHAPTER SUMMARY

- In this chapter we looked at various performance optimization techniques.
- Content peformance optimization best practices includes Optimized publishing workflows, database-level performance optimizations, content caching design, monitoring and notification setup, proactive identification, of memory leaks, CMS-level performance optimizations, Omni-channel content strategy, CMS caching, search engine level performance optimization, infrastructure level performance optimization
- The generic content best practices are Separation of content and presentation, Optimized digital assets, Standards-based implementation, continuous content improvement, Content development, The main characteristics of good content are Usability, Customer-centricity, Simplicity and clarity, Messaging strategy, Conciseness, Findability and searchability, Contextual and relevance, Consistency, Accessibility and Omni-channel enablement, SEO friendliness, Content readability, Content messaging and communication, User engagement and user adoption, Key Content best practices are Content chunk strategy, Deliver personalized and contextualized content, Create user-centered content, Optimize content ecosystem, Adopt content reusability strategy, Identify and eliminate duplicate, unnecessary, and outdated content, Intuitive information architecture and information discovery, Content tracking and optimization, Content presentation.
- The key CMS best practices are Robust requirements elaboration, Business-friendly processes, Standards-based platform, Open architecture, Minimization of content silos, Centralized content access, Content governance, Metadata-driven content discovery, Reusability, Content services, content model and content template design, optimal workflow design.

Part 3

Enterprise Search Technologies

Chapter 13

Introduction to Enterprise Search

Today we live in a search-dominated world. Search forms the primary gateway for our online information exploration, influence our decisions, shapes our shopping experience, enriches our learning, and completes our actions in numerous ways. A search function is more than a "question-answer" engine; it organizes information and provides us with sensible and relevant information.

Even in an enterprise scenario, search plays a major role in information discovery. Due to exponential growth of enterprise content and rise in the volume of unstructured data, enterprises face huge challenges in managing the data and making sense out of the data for business and end-users. Search helps organizations realize the full value of the information. For enterprises, search is an enabler for creating differentiated user experiences, and it plays a vital role in achieving long-term user engagement. Search can engage enterprise users in more than one way. It can initiate the crucial dialog with first-time visitors by providing them with relevant information, but it is also a quintessential business enabler, creating a robust information retrieval and self-service platform. It opens up a host of business opportunities to engage with partners and users at various levels.

In this chapter we take a look at various aspects of enterprise search. We first examine the main motivations, challenges, and best practices for enterprise search. In the next section we look at layered search architecture and search capabilities. In the concluding section we have a detailed discussion of enterprise search features. Specifically, we take a look at the general features of Apache Solr and ElasticSearch. The chapter lays out the foundation for the upcoming chapters in this part of the book.

Enterprise search architects, information architects, enterprise architects, and search developers will find this chapter useful.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

13.1 INTRODUCTION TO ENTERPRISE SEARCH

Enterprise search is a key tool in the information retrieval process and plays a key role in organizing enterprise information. Search acts as window to enterprise content by providing relevant views of information. Information is a critical asset for organization and is present in various formats such as Web pages, portals, services, e-mail, documents, collaboration, and CMS content. Search should be able to make sense of all structured and unstructured information within an enterprise. Enterprise search can "weave" relevant enterprise content and connect the dots in the enterprise data to provide a holistic dashboard view. Through enterprise search we can search the relevant data, infer patterns, build intuitive visualizations based on obtained data, follow related social conversations and discussions, and map resources required for a specific task. This unified presentation of relevant information ultimately benefits all audiences, who can get more out of the underlying data. Enterprise search can be plugged into crucial processes to make them more efficient. For instance, search could enhance content reusability by discovering content and assets in a content-authoring process; search can also help automatic routing in a workflow by finding the person with right skillset. Enterprise search opens up numerous possibilities to optimize the business processes and user experience. Some of the related standards for search-based information retrieval are ANSI/NISO Z39.50 Search and Retrieval Protocol (ISO 23950) (standard protocol is for searching and retrieving electronic information over a network) and metadata standards (such as Dublin core ISO 15836 and W3C's RDF). In this section we take a look at the search drivers, tenets, challenges, and best practices of enterprise search.

What compels business and technologists to use search technology? In the upcoming sections we present key business scenarios and use cases for search. Before going there, however, let us look at the main business and technology drivers for enterprise search.

Business Drivers

- **Efficient discovery of relevant information:** Search engines index numerous enterprise sources and thus can help in dynamic discovery of relevant information. For instance, a dealer can search for customer information, and an efficient search engine can provide all conversations (such as chat information, blogs, posts) and interactions by the customer aggregated from a collaboration platform (for UGC), user repository (for user profile information), incident management system (for incident data), and the like, and this provides a 360-degree view of customer activities for a dealer. This information can be used to provide personalized offers and recommendations. Personalized search is one of the search flavors that would provide search results based on user preferences (specified explicitly or derived implicitly).

- **Efficient usage of enterprise information:** Organizations need to effectively use all the information available to them. Search can help business in organizing and structuring information.
- **Information reuse:** Search is an efficient tool to dynamically aggregate enterprise information and reuse the existing information. It eliminates content silos and reduces duplicate content creation.
- **Search-centered experience and delivery:** Due to numerous applications of search, many digital platforms are designed around search and position search as a key information discovery and navigation tool.
- **Self-service enablement and productivity improvement:** Search can also enable self-service capability for an enterprise. For instance, in an incident management workflow, search can suggest relevant solution to the user before he or she logs the incident. If the solution can be addressed by the solution provided by search, it can avoid the incident/support call. Similarly, search-powered product recommendations, dynamic navigation provided by search, and dynamic aggregation of relevant topics (such as popular product downloads, most accessed content) may serve the business's goals such as self-service, cross-sell, and upsell. An efficient search platform can make the users' job easier and help them complete the tasks faster.
- **Personalization and targeted content delivery:** Users often waste lot of time in finding the relevant information, and abandon the site if they are not satisfied with the obtained information. Thus, quick discovery of relevant information becomes a key success factor in information management. Successful business strategy and business outcomes (such as conversion rates, order values, site traffic, loyalty, return customers, customer satisfaction, cost optimization, reduction in operational cost) are directly dependent on how fast a digital user can discover the required information. The ease and speed of relevant information discovery directly impact the outcomes.
- **Aggregation platform:** Businesses need a contextual information retrieval and dynamic information aggregation platform. Search is well positioned to perform a dynamic aggregation of relevant data to provide a unified view.
- **User influencer:** An effective search with relevant information, effective information discovery, and guided navigation can significantly influence user's purchasing decisions.
- **User engagement and collaboration:** Requirement of an active and deep user engagement platform with increased site stickiness can easily attract new customers. Business's interest is in developing a long-term relationship with existing customers and increasing loyalty through differentiated offerings. Businesses need to have a continued "conversation" with end-users for a long-term engagement. Search platforms may provide a solid backbone for user engagement platforms. Search may also facilitate collaboration through people search and content search.

- **Competitive advantage** consists of the ability to move to the next-generation digital platform to provide more contextualized, relevant, and complete information through search.
- **Business promotion:** Businesses promote and expand the self-service model to reduce operational and overall business costs.
- **Business growth and online revenue:** Organizations can easily manage information growth required for their business through search. Using search, enterprises can enhance business opportunities, ROI, and conversion rates through highly relevant/personalized recommendations and cross-sell/upsell packages.
- **Unified information access:** Businesses need to leverage existing assets and provide unified information access for all enterprise information. A search engine can connect to various structured and unstructured enterprise sources and index content in various formats to provide a unified view.

Technology Drivers

- **Information discovery:** This is the most important and natural scenario for search wherein it can provide the most relevant and personalized content/data for the end-user based on relevancy algorithms. Exponential growth in enterprise content volume and an increasing number of enterprise content sources pose challenges in information discovery, and enterprise search can fill this gap.
- **Dynamic content aggregation:** Search can act as a platform and a front-end portal by providing navigation, personalization, and content aggregation.
- **Content reuse:** Existing content can be reused in multiple applications through the search platform. For instance, a product description content can be used on the main e-commerce site and other micro-sites and Web applications by surfacing the same content in a search result.
- **Information organization:** Businesses have a critical need to structure and organize diverse enterprise content and digital assets with uniform classification and categorization, as well as to organize structured and unstructured enterprise information.
- **Social indexing:** To succeed, businesses aim to index social media and user-generated content requiring big data search capabilities. Search can help monitor and report social conversations and perform sentiment analysis.
- **Activity automation:** Businesses have a need for technologies for automatic content categorization, automatic request routing, and automatic notification.
- **Context-based delivery:** Businesses have a need to provide personalized navigation and recommendations based on user context.
- **Platform approach:** Businesses aim to provide the ability to extend the platform through search services and search APIs.

- **Semantic analysis:** Semantic search platform can interpret natural language-based long-tail queries to provide the most relevant results.

In next section we will look at common challenges we face in enterprise search.

Challenges of Enterprise Search

- **Poor relevancy in search results:** This could be due to various reasons. The most prominent are:
    - Incorrect configuration or implementation of search relevancy algorithm
    - Incomplete indexing of all enterprise content sources, leading to incomplete search results
    - Incorrect transformation of index content into uniform format and incorrect content classification/categorization methods
    - Lack of content consistency and classification capability.
- **Search results containing outdated content:** This is mainly due to crawling and indexing frequency and not considering content update timestamp for ranking search results.
- **Incorrect linking to documents:** Search engines often need to transform the relative path into a correct URL or absolute path. If the URL conversion is wrong, then it leads to broken links.
- **Security issues:** Sometimes search algorithms do not filter the search results based on requested user's security permissions, leading to accidental disclosure of confidential information.
- **Lack of metadata and SEO tags:** If the enterprise content is not tagged with relevant metadata, even structured documents may pose challenges for search.
- **Search performance with a large dataset:** As the enterprise content volume grows, some search algorithms face issues in scalability and performance (especially if the index servers are not clustered). Same issues may surface while indexing large volumes of data such as social media content or UGC content.
- **Unstructured content volume:** Enterprise content is a mix of structured and unstructured data. A rapid increase of unstructured content in enterprise systems poses challenges to search engines in terms of efficient indexing, transforming, categorizing, and calculating relevancy. Unstructured information contains various content formats and lacks structure, which makes the job of a search engine more difficult. If unstructured data were not properly crawled, indexed, and categorized, then it would be hard to make sense of customers' conversations and difficult to get the complete picture of their needs and preferences.
- **Data fragmentation:** Often enterprise data is distributed across various systems and channels. Search engines have to mine the data and establish a

relationship between various parts of it to come up with relevant search results. For instance, in an e-commerce platform, product data would be distributed across database (product attributes), CMS (product information content), pricing system (product pricing and availability), and social platforms (product reviews, comments, and rating); a search engine has to collate the information from all these systems for a given product to provide a unified view of the product information (like a product dashboard page).

- **Various interaction channels:** Modern enterprises provide various channels (such as Web, mobile, social, phone, kiosk, stores, wearable, etc.) to engage with customers. This poses challenges to search engine to obtain a holistic view of all customer interactions across various channel touch points. The search presentation should also be optimized across all channels.
- **Digital asset search:** A search engine should be able to index various digital assets such as audio/video files, images, binary documents, etc. If a search engine is unable to index the assets, it may be difficult to provide meaningful search results.
- **Lack of business controls:** Each business scenario has a different context. An enterprise search engine should provide controls to configure search parameters such as relevancy ranking, rank boosting, content sources, etc. If a search engine does not provide proper controls to business stakeholders, it affects business outcomes. Business stakeholders may need controls to fine-tune relevancy criteria and other elements for product promotion.
- **Challenges in crawling discrete data sources:** Some search engines do not have connectors to crawl and index discrete content sources such as ERP systems, legacy systems, etc.

In next section we will look at proven best practices while implementing search platform within enterprises.

Generic Best Practices of Enterprise Search

- It is recommended to cleanse and structure content that is in our control. For instance, CMS-generated content can be tagged with metadata, should be structured, and should be in a valid format (HTML, PDF, etc.). This helps search engines index it better. Similarly, documents from the document management system and assets managed in the digital asset management system should be appropriately tagged and categorized.
- Implement SEO guidelines by providing accurate page-level keywords, metadata, title values, and alt text for media files.
- Ensure that all key pages are reachable through direct or indirect links from the main pages. If there are disconnected or orphan pages, provide a seed-list page (linked from the home page or from sitemap) that links to all orphan pages.
- Specify crawling restrictions in robots.txt.
- A search engine should be able to crawl a wide variety of enterprise content sources such as content management systems, file servers, database servers,

ERP systems, Web site content, document repositories, etc. In addition, it needs to index unstructured enterprise content such as chat, e-mails, blogs, audio/video, feeds, files, and the like.

- Size the search infrastructure to support the current and future content growth. It is always recommended to use a clustered configuration for index servers.

13.2 ENTERPRISE SEARCH OVERVIEW

In this section we present a broad overview of a search platform. We look at the components of a typical enterprise search platform and then look at some of the emerging trends and the evolution of enterprise search technologies.

Enterprise Search Architecture Layers

In this section we look at the general components of an enterprise search engine. The main considerations for designing each of the search layers are also discussed. A detailed architecture of an enterprise search will be presented in the next chapter. A typical enterprise search platform is shown in Figure 13.1.

We can broadly identify three layers in an enterprise search engine platform.

Search user experience layer

Search UX is aimed at providing easy-to-use search interface and targets to provide relevant information discovery. It also offers user engagement features by providing dynamic faceted navigation, filtered search, guided search, personalized search, and expression support. The main design considerations for this layer are intuitive user

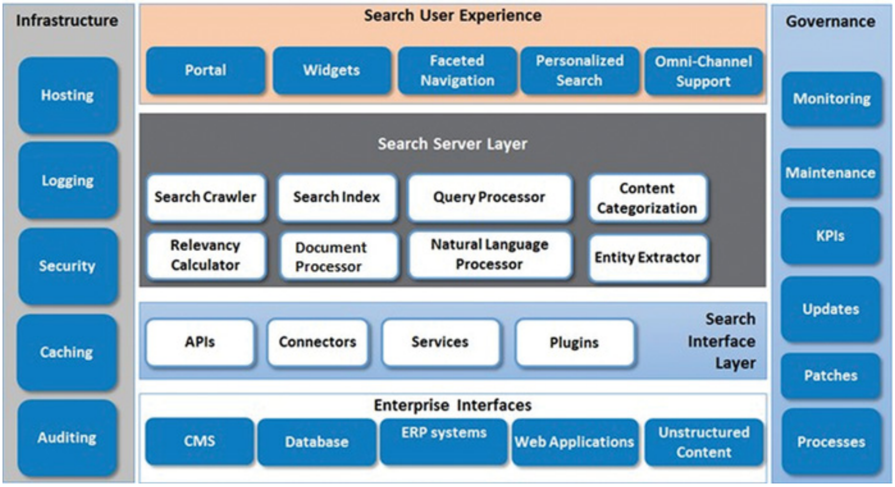


Figure 13.1 Enterprise Search Platform

experience design, optimal page performance, Omni-channel support, and internationalization support.

Most enterprise search engines provide a built-in search portal, which can be customized based on individual needs of the business. The search portal provides user experience components such as search tool, search results page, and advanced search interface. The basic search function provides basic keyword-based search and other user-friendly features such as type-ahead search. The search results page provides ranked search results (based on criteria such as relevancy, modified date, etc.) in a paginated form. An advanced search function provides features such as synonym search, spell-check, faceted search, inclusion/exclusion of keywords, and many others.

Search server layer

This layer consists of core search components including search crawler and indexer. Search crawlers crawl and index the content. Search engine components consist of query processor, semantic/natural language processor, multilingual processor, caching components, and others. We will look at the complete search process in the upcoming sections.

The main design considerations for this layer are indexing design (partial and full indexing), relevancy ranking configuration, index support for various content types, and scalable design (to support excessively large content volume). Search administration interface can be used for various configurations such as starting URL and for fine-tuning relevancy ranking.

Search interface layer

The function of an interface layer is twofold: connect to various content sources to index the content, and expose search services for external consumers. Enterprise search engines provide various built-in interfaces such as search portal (a user interface for end-users and administrators), APIs (for programmatic access to search functions), and search services (services access to search functions). The main design considerations for this layer are flexible standards-based services design, connector design, and search API design.

Search infrastructure (auditing, logging, hosting, caching) and search governance (maintenance, monitoring, update, KPIs) span across all search layers. We will take a more detailed look at the search features in the upcoming sections.

Web Search vs. Enterprise Search

This book focuses mainly on enterprise search for digital platforms. This should not be confused with the Web search. Table 13.1 depicts basic differences between enterprise search and Web search.

Table 13.1 Enterprise search vs. Web search

Category	Enterprise Search	Web Search
Ranking algorithm	The ranking is predominantly based on determining relevance of enterprise content and documents through metadata, user context, keyword, tags proximity, document freshness, keyword density, etc.	The Web search results ranking is predominantly based on link popularity, quality of inbound links, random walk algorithms, page ranking, and page/domain-level keywords.
Security	Enterprise search indexes both secured and unsecured content and filters the search results based on role-based access for the authorized user.	Web search mostly crawls and indexes public content that can be accessed by the general public.
Content format	Enterprise content format is variable, with various formats and types; there is minimal interlinking across documents and content. Enterprise search indexes heterogeneous enterprise content such as HTML, documents, ERP, binary assets along with structured (database, files), and unstructured content (e-mail, chat) from various enterprise content sources.	Webs content is mostly HTML and is generally interlinked. A Web search engine mostly crawls Web documents and other structured documents in public domain. The content format is predominantly HTML.
Search features	Enterprise search supports a wide variety of advanced search features such as faceted navigation, relevancy configuration, entity extraction, search filtering/sorting, etc. Normally the user experience will be more advanced than in Web search.	Web search mainly supports synonym search and Boolean operators as its advanced search features. User experience is simple (generally contains a single search box on the page).
Information relationship	The data and documents are mostly compartmentalized with minimal relationship between them.	Web documents are highly interlinked, and often incoming links are used for calculating relevancy.
Focus on business verticals	Enterprise classification platforms (such as domain-specific taxonomy) provide a structured definition of business-specific terms (including synonyms, relationships, etc.). Enterprise search uses this to customize the search process for the corresponding business verticals.	Web search is a general-purpose search engine for public Web documents. It does not differentiate business functions.

(*continued*)

Table 13.1 (*Continued*)

Category	Enterprise Search	Web Search
Content quality	Enterprise search engines also consider duplicate removal, security attributes, and user context to provide high-quality content results. Often enterprise searches also aggregate content from external sources through federated search.	Web search mostly uses search ranking algorithms to produce quality search results.
Search controls	Enterprise search provides finer controls such as performance monitoring, configuring relevancy ranking, rank boosting, business-related synonyms, content promotion, etc.	Web search offers little admin-level controls for the end-users.
Search query	Always associated with context (such as user attributes, roles, job, security, site section, language, geography, purchase history, access device, etc.) along with regular search terms.	Predominantly uses only search terms. A small percentage utilizes user's attributes to provide personalized search results.

Search-Related Trends

Search technologies are quickly assuming various key roles in a digital ecosystem. We have noticed the following trends in enterprise search space:

- **Big data search:** As organizations are facing challenges in organizing big data, search technologies are leveraged to extract the relevant content from big data. There is an increasing convergence of big data, text analytics, business intelligence, and data-mining technologies with search technologies.
- **Search-based knowledge platforms:** Though search always was an integral part of knowledge systems, search is now being used in broader scope in knowledge management. Search capabilities such as semantic analysis, natural language processing, automatic topic clustering and classification, relationship mining are increasingly being used in knowledge platforms. Search can combine relevant pieces of knowledge content with assets to provide actionable knowledge. We could easily integrate information from other knowledge sources through federated search.
- **Mobile search:** Moving away from keyword search, mobile search mainly uses voice queries, gestures to provide actionable information. Mobile

search adjusts the relevancy for mobile access (by considering location awareness), providing mobile-friendly user navigation, predictive search, providing search services/APIs for building mobile applications and such activities.

- **Answers, Not search results:** Enterprise search engines would use relevancy factors to determine the most relevant answer for the search query. Answer would then be provided in response to search query instead of result list.
- **Personalized Search:** Delivering most relevant content based on user attributes could further enhance relevancy of search results. User's preferences, location, device, transaction history would be factored in while providing personalized search results. Search engine would provide a holistic view of user data through aggregation of results from all internal enterprise systems and social platforms. Users could also personalize their search experience.
- **Federated search:** Due to increase in complexity of enterprise eco-system, federated search is needed to get information from various search interfaces, repositories and to present holistic view of the aggregated information. The main federated search engines send search queries to various search interfaces and aggregates search results.
- **Search based applications:** Many enterprise applications use search as their main technology backbone. Workflows, process automation systems (using search driven routing), voice of customer (VOC) systems (using real-time cross-channel federated search), research platforms, customer support platforms (by using search based relevant and actionable information discovery), data warehouse applications, e-commerce applications (through searchandising options leading to increased conversions), business Intelligence platforms etc. primarily use search along with enterprise taxonomy for implementing their main features. These applications use dynamic aggregation, unified content access, connectors, and features of such along with domain vocabularies in enterprise taxonomies.
- **Focus on open-source software:** Enterprises are looking to leverage the open source software and standards wherever possible. This involves customizing open source search for enterprise needs to building enterprise-grade scalable platform.
- **People search:** Enterprise search is also increasingly used for search people based on their expertize/skill set from unstructured data formats such as email, source control systems, project reports and such.
- **Semantic search:** Users expect contextually relevant results personalized for their scenario. This requires deeper understanding of search query keywords, their context along with natural language processing techniques to get the overall "meaning" of search terms. The quality and accuracy of search results are improving due to vast advances in relevancy ranking algorithms.

- **Software-as-a-service (SaaS) model:** Many enterprise search vendors are providing cloud hosting and software as service model for enterprises. Security and privacy are some of the key concerns in this area.
- **Business Intelligence (BI) capabilities:** We are noticing increasing trend in convergence of BI and search technologies. Through unified information access and unstructured data analysis, search technologies can drive BI dashboards, reports and visualizations. Search could augment and compliment BI technologies through processing of unstructured information and relevant information discovery.
- **Search as Unified Solution platform:** Search could integrate desperate enterprise sources to obtain and process relevant information from structured and unstructured sources. Enterprise search should process various content formats to provide relevant results. Leverage search as a solution platform for dynamic delivery systems, business intelligence systems, information management systems, e-commerce and other business verticals. Search can act as a front-end component or as unified information discovery portal. Using the core search features exposed as services and APIs, we could extend the platform and allow other systems to use those services.
- **Search driven analytic visualizations:** Analytics reports such as location trends, sales trends are powered by search.
- **Search analytics:** Monitoring search usage through search analytics is used to enhance overall information retrieval and presentation process. Search analytics would provide insights into usage of search results, information usage across geographies, user information needs and user challenges.

We will be having detailed discussion on some of these topics including federated search, semantic search, personalized search, open source search engine, people search and Big data search in coming chapters.

Search Evolution

A high level evolution of enterprise search is depicted in Figure 13.2

- Keyword search (basic search): Search algorithm does a keyword match for determining relevancy. This was the first phase of search.
- Relevancy Ranking and Advanced search: Search engine provides configurable relevancy criteria based on enterprise needs. Search administrator can add criteria such as link importance, content freshness, access frequency and such for determining relevancy. This includes advanced search capabilities such as synonym search, federated search, faceted navigation etc.
- Semantic search: In this scenario, the search algorithm extracts entities and topics which conveys semantics of the search terms and search happens on the “meaning” of search terms.

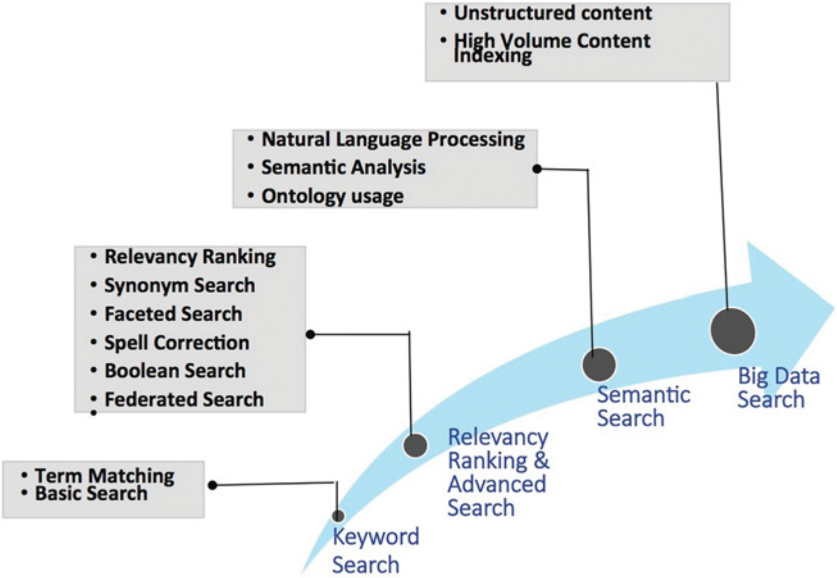


Figure 13.2 Enterprise Search Evolution

- Big Data search: This category of search involves searching large volume of data. Data is mainly unstructured consisting of blogs, text, social posts, chat etc.

Key Value Proposition of Enterprise Search

Enterprise digital Search enables multiple digital capabilities and hence it is one of the core elements of a digital strategy. Given below are some of the main value propositions of enterprise search categorized in business domains.

Table 13.2 provides sample business scenarios across various solution domains where search can be used effectively:

13.3 ENTERPRISE SEARCH CAPABILITIES

An enterprise-wide digital search system should be well equipped to connect to various enterprise systems and index content of various formats so as to provide meaningful search results. Let us look at some of the core search capabilities for an efficient digital search platform. Though the capabilities of a search system vary somewhat across industry verticals and are based on business domains, the core capabilities generally remain the same. Listed below are some of the core search capabilities.

Table 13.2 Search benefits for business domains

Business domain	Search potential benefits
Retail	• Holistic multi-channel search experience across all channels. • Boosting online conversion rates, multi-channel sales through efficient discovery of information and by providing highly relevant results • Search can use user segments to provide more insights into customer's interaction with various kinds of content. • Search can increase site traffic and return customers through search-driven personalization/recommendations and search-based target content • Search can reduce total cost of ownership and operational costs through search-based self-service features • Search can increase customer loyalty through personalized search results based on their interests and by providing more accurate and relevant search results along with search-based navigation • Search-driven advertisement and product/content promotion can be used for increased monetization
Ecommerce	• Search-driven store fronts power ecommerce operations • Search-powered product promotions • Search could provide exceptional user experience through search based product comparison tools and other decision making tools • Search can provide a differentiated shopping experience and a holistic view of product information (such as user reviews, product rating, user guides, external reviews) • Provides search-driven personalized product recommendations and navigation to enhance cross-sell and up-sell opportunities
Publishing industry	• Search would provide relevant content to all stakeholders • Search integrates with all external partners and third-party content sources to make the search results more meaningful and accurate • Search connects readers to relevant content and index user generated content (UGC) such as blogs, reviews, and internal social platforms to include UGC in search results.
Intranet systems	• Intranet search provides relevant content and documents (such as how-to documents, internal CMS content, internal site search etc.) for internal stakeholders such as employees, administrators, business stakeholders etc. • Security is one of the prime concerns. Search should filter the results based on role-based access.
Business Intelligence	• Search-driven data visualizations • Search-based analytics reports • Search based BI dashboards

(*continued*)

Table 13.2 (*Continued*)

Business domain	Search potential benefits
Knowledge and collaboration domains	• Search will be quintessential to retrieve the relevant documents from knowledge repositories • Search promotes self-service by harnessing the collective intelligence in collaborative platforms. For instance, suggested solutions, most popular downloads, top-rated discussions promotes self-service model for end users • Search enables customer service representatives to efficiently use the enterprise information to solve user queries quickly

- **Ability to crawl and index various content formats:** The search engine should be able to index documents, content, and data types of various structures and formats. Additionally, the search engine should be able to index and process data and content in an unstructured format. Normally search engines convert and cleanse the crawled data into a structured document model for easier indexing and processing.
- **Business-tailored relevancy:** A search process should provide results that are relevant to the search query keywords and context. While each search product uses different algorithms for calculating relevancy, the most commonly used techniques utilize keyword proximity, keyword position, and other similar criteria. The relevancy ranking of search results needs to be altered to suit the business needs. This includes rank boosting, product promotion, and altering weightage for factors influencing ranking (such as content freshness, access frequency, content rating, etc.). Search administrators and business stakeholders should be able to configure the high-level parameters used for relevancy ranking.
- **Support for filters:** A search engine should provide various filters such as document type, document size, site URL, document rating, etc. in an "advanced search" feature to filter search results. Additionally, some search engines support SQL-type queries as an advanced search feature.
- **Support for search personalization:** Search engines should be able to filter the search results based on implicit values (such as user permissions, applicable roles, data access policies, etc.) and explicitly specified user preferences. Other variants of search personalization include, among others, saved search results, personalized search-based navigation, and personalized search-based recommendations.
- **User interfaces for search management:** Search engines should provide a business-friendly user interface for management purposes. Search administrators and business stakeholders should be able to use this interface to administer search.

- **Extensible and pluggable architecture:** Extensibility is relevant for integrating a search engine into new systems. A search engine should provide extensible plugins and integration architecture so that it is easy to develop new adaptors for integrating with newer systems. When external systems or third-party interfaces want to consume search, the search systems can provide search plugins such as search widgets, search portlets, or search web apps.
- **Search interfaces:** The search results should be consumable from other third-party systems through services, APIs, feeds, and pluggable widgets. Search systems should expose SOAP or REST services and APIs for external systems and consumers so that they can develop search based applications such as mashups and others.
- **Semantic support:** A search engine should be able to provide linguistic and semantic support through synonym-based search, concept/entity discovery, related terms/searches, sentiment analysis, linguistic processing, etc.
- **Support for faceted navigation and sorting:** A search engine should be able to discover the content structure (for instance, in e-commerce, a search engine should be able to discover product attributes such as price range, model, vendor, etc.) and construct a filtered navigation based on this. This provides the user with a predefined set of filters to easily discover the relevant product and information. It should also support guided navigation, synonym support, concept clustering and categorization, and other advanced features.
- **Scalable platform:** The search platform should be able to provide a scalable architecture to index large amounts of ever-increasing enterprise data and provide search results in an intuitive user interface. On average, regular search technologies should be able to index and scale up to 1 TB (terabyte) of enterprise data. Data beyond 1 TB would be considered big data (*Note: The size for classifying big data is a fluid category. As the search and processing technologies improve, the threshold for categorizing big data may also increase.*)
- **Big data and social search:** Search platforms should be able to index excessively large amounts of enterprise content and social content to provide meaningful results.

In addition to this, specialty search engines provide domain-specific features such as social indexing, sentiment analysis, big data indexing, etc.

13.4 ENTERPRISE SEARCH FEATURES

In this section we will look at various features of enterprise search.

Basic Search Features

Keyword search

This is part of the basic keyword search in which users provide the search term as input criteria. Generally there are three types of keyword search: “allword” (which provides search results containing all input search terms), “anyword” (which provides search results matching any of the input search terms), and “exactphrase” (which provides search results matching the exact search phrase). Normally “exactphrase” search can be conducted by embedding the search phrase within quotes, and the search does a proximity match for the specified search terms. Some search engines also provide the option to exclude specified keywords while matching (using “exclude” or “not including” feature).

Dynamic navigation search or faceted search

A search facet is a logical category or a property/attribute for the indexed content. Search engines can “auto-discover” content categories or use enterprise taxonomy for identifying categories. Facets can be obtained from the content category, content metadata or page meta-tag values, or entity attributes. In product search, product model, price range, and product features could become facets.

In faceted search, each record in the result set is a part of one or more categories defined in the enterprise search engine. Each category (navigator/facet) contains a list of filters (modifiers) to refine the search. In faceted search, search results are grouped under various facets. Users can select the facet to further filter the search results.

A simple example of facets in product search includes product brands based facet and price range based facet.

Sorting

This feature allows the end-user to sort the search results based on their attributes. Normally the search portal or search-user interface will provide the sorting feature. In some search engines we can specify the default sort criteria along with the search term. Search interfaces also provide an option for the end-users to sort the results based on individual columns.

Pagination

If the result set contains a large number of results, those will be displayed in a pagination mode. Pagination helps in better query performance and page performance.

Advanced Search Features

Advanced search is used when users know the exact search phrase or when users need to filter the results in more complex ways.

Boolean search

Users can specify the Boolean operators such as AND, OR, and NOT along with search keywords. Some search engines also support the symbols such as “&”, “|“,

"∼" instead of the regular Boolean keywords. A search engine matches the relevant documents based on the input combination.

Synonym support

Many search engines provide configurations to support synonym search. In such cases the search results will include the words that have same meaning as the specified search term. Synonyms increase "recall" value of search relevance. Synonym support can be enabled through the search engine configuration and it can be specified while sending the search criteria.

Sometimes search admins can also define and update the synonyms for business-critical words in the search administration interface and define custom synonyms wherever necessary.

Spell check / Did you mean (DYM)

Search engines automatically correct the misspelled search keywords and provide the search results for the correct word. Alternatively, a search engine can prompt the appropriate word or correct search term through the "*Did you mean*?" feature. "Did you mean" uses the closest matching index word for a given search term through a best guess. Search administrators can configure the "Did you mean" option for various languages.

Anti-Phrasing and Stop Word

The anti-phrasing or "stop word" feature of the search engine removes the noise from the search terms by removing common phrases like "who is", "what is", "how do I", and "where can I find" from the query to have more precise searches. These phrases are defined in the anti-phrasing dictionary. Search administrators can also configure any business or organization specific stop words. Generic list of stop words include "a", "the", "of", "it", "to", "on", "by", "with", "that", and "this".

Wildcard search

Users can specify the keyword pattern instead of specifying the exact keyword. This feature is used when users are not sure of the exact search term or want to explore a wide variety of topics. For instance, the search term "prod∗" provides all search results that contain the term starting with "prod." Similarly, we can specify the ending keyword, contains keyword, and others using wildcard.

Lemmatization and Stemming

Lemmatization and stemming allows the searching of different forms of the same word. For instance, "acting" is a form of the base word "act" (known as lemma). In lemmatization the search engine "expands" a base word (lemma) into all its forms for searching, and in stemming the search engine “contracts” the query word into its base form; when the search terms contains “acting,” the search engine searches for documents that contain “act” (which is obtained from stemming “acting”).

Featured results

Many enterprise search engines provide an option to feature content for a specified set of keywords, in addition to the regular search results (organic search results) for the search term. Business stakeholders can use this to promote business-critical content, offers, and promotions; they can also be used to draw the users’ attention to the recently launched products, campaigns, and other similar content. Business stakeholders often configure the featured results using the search administration interface.

Multi-field search

In this feature, a search request can specify the keywords for a specific index field. For instance, we can specify: “Provide me all documents in which *title* field has *User Interface* and *data modified* field is *within last month*” Title Has User Interface (exact phrase) AND data modified < 30 days

In this example, a search engine has to match the keywords in two separate fields.

Type ahead

This is a very prominent feature of enterprise search engines. Type ahead is an interactive feature that provides the keyword suggestions when user starts typing a search term into the search interface. One of the variants of this feature is to provide a categorized list of keyword suggestions. Most of the search engines accept a partial keyword match feature (which offers suggestions as soon as the user begins typing) and will also consider other context parameters such as user’s role and application context while providing the keyword suggestions.

Advanced security features

Normally the user context (such as user profile attributes, roles, and permissions) is used to filter the search results to ensure that user is provided with only the authorized content. In addition to this feature, we can create a security profile and map the Web applications or client applications to this security profile. (Similar to the way we map the users to a role). This security profile in turn restricts the search results from a particular search collection or a data source based on security restrictions. This ensures that applications and users do not “see” any unauthorized or confidential information for which they do not have access.

Here is an example of three applications: employee portal, supplier portal, and business portal. Out of these three applications, business portal has access to highly confidential organization information such as business leads, opportunity pipeline, etc., and is stored in a separate ERP. Obviously some of this information should be restricted for internal and external users. As enterprise search engine indexes all of the enterprise data sources, we need to ensure that content is shown only to authorized users. One way to ensure this is to create separate search collections for each data sources and map the security profile to those search collections. In this example, there can be a separate search collection for business portal ERP and a separate security profile mapped to this collection. When the request comes from a business portal application, it will be given access to content indexed from the business ERP according to its security profile. In same way we can restrict access to this collection to employee portal and supplier portal applications.

Features in Apache Solr and ElasticSearch

So far we have seen various features of an enterprise search engine. Let us now look at two popular open-source search engines, Apache Solr and ElasticSearch, and explore the features they offer.

Both Apache Solr and ElasticSearch use Lucene for core search features. Some of the features supported by these two products are listed in Table 13.3.

> *Note: The features listed in Table 13.3 are for educational purposes only. The list is partial and is not an evaluation or comparison or ranking of search products.*

Both search engines support many core search features such as synonym support, dynamic fields, faceted search, spell-check, auto complete, results highlighting, and others.

Table 13.3 Apache Solr and ElasticSearch Features

	Apache Solr 5.3.1	ElasticSearch 2.0.0
Access method	Supports Java API and REST-based access	Supports Java API and REST over HTTP
Search query	URL query with parametersSupports field search and other advanced search query features	Supports Search Lite (URL query) and Query DSL (multi-level JSON structure)
Relevancy calculation	Uses weighted scoring relevancy model	Uses similarity algorithm that factors term frequency, inverse document frequency, and field length form
Importing data	Can be handled via DataImportHandlers for import data from files (XML, CSV), JDBC data	Uses rivers module to import from a variety of sources such as database, JMS, RSS feeds, etc.
Scalability	Support SolrCloud along with Apache Zookeeper for distribution	Master-slave configuration provided for scalability on ElasticSearch nodes

13.5 CHAPTER SUMMARY

- The chapter introduces various aspects of enterprise search.
- The key business drivers for enterprise search are targeted content delivery, dynamic aggregation platform, user influencer, user engagement, competitive advantage, promotion and self-service, revenue, and unified information access.
- Main technology drivers for enterprise search are information discovery, dynamic content aggregation, content reuse, information organization, social indexing, context-based delivery, and platform approach
- Key challenges in enterprise search are poor relevancy in search results, search results containing outdated content, incorrect links, security issues, performance issues, content volume, data fragmentation, multiple channels, asset search, business controls, and discrete data sources.
- Main layers of search platform are search user experience layer, search server layer, and search interface layer.
- Web search is mainly a general-purpose search engine, whereas enterprise search caters to a specific business domain and indexes enterprise content.
- The key emerging trends in enterprise search are search as platform, mobile search, accuracy in search results, federated search, open-source software, search services, semantic search, SaaS model, and BI support.
- Various phases of search evolution are keyword search, advanced search, relevancy ranking, and semantic search.
- Various capabilities of enterprise search are ability to crawl and index various content formats, business-tailored relevancy, support for filters, support for search personalization, user interfaces for search management, extensible and pluggable architecture, search interfaces, semantic support, support for faceted navigation and sorting, scalable platform, and big data and social search.
- Various features of enterprise search engine are keyword search, dynamic navigation search or faceted search, sorting, pagination, advanced search features (Boolean operator support, synonym support, spell-check, anti-phrasing, wildcard search, lemmatization, featured results, multi-field search), and advanced security features.

Chapter 14

Advanced Enterprise Search

Enabling search capability for enterprises with a complex technology ecosystem requires careful planning and considerations. We would need to engage all concerned stakeholders to design for optimal integrations; business administrators will need to test the built-in relevancy parameters and adjust it to suit business needs; security has to be enabled while crawling protected resources; federated search needs to be implemented to aggregate search results from enterprise sources; and so on. We will build on those concepts to elaborate some of the advanced features in enterprise search.

In this chapter we take a look at advanced topics of search. We begin with a discussion of federated search along with its design topics and architecture. We then look at advanced search features such as relevancy rank adjustment, personalized search, and alternative search suggestions. Next we discuss secured search and underlying security integration. A detailed discussion of semantic search will follow, along with its architecture and semantic search process. In the subsequent sections we take a look at people search, social search, and big data search. Finally we look at a search case study of a content and document management portal driven by Apache Solr search engine. This will help us understand the applicability of search in various scenarios.

Search architects, enterprise architects, program managers, and search developers will find this chapter useful.

14.1 FEDERATED SEARCH

In some complex enterprise scenarios, digital ecosystems comprise of multiple information systems (possibly with incompatible interfaces). For instance, an enterprise might contain multiple platforms built on different technologies, each with its own search engines and services. In some cases, a digital application might need to aggregate information/search results from third-party information systems having incompatible search interfaces (like non-indexable content, absence of search services, etc.), or digital applications need to leverage search services of external systems.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

For instance, let us consider a product aggregation platform that lists the availability and pricing information of products from various internal and external systems. In this case we not only need an enterprise search engine that provides the information for the specific product from internal product catalogs; we also should obtain the information from external product catalog systems. In such cases, federated search comes to our rescue. Another classical use case for federated search is creating a unified book catalog system that needs to search for books from various external repositories and collections. *Federated search system aggregates search results from multiple information systems through parallel search query execution.* Federated search systems use connectors, services, APIs, feeds, and other integration methodologies to provide holistic and unified view of search results from various information systems.

Using federated search we can submit a query term to various internal and external information systems (preferably with virtual index). This would help us leverage the existing information retrieval systems and their search infrastructure without needing to index or crawl all the sources (thereby incurring associated hardware costs) and help us provide more detailed and complete search results for a query. Federated search also helps launch a dynamic delivery platform quickly by plugging into existing information retrieval technologies; more often we just need to create a user interface to show the aggregated search results.

Features of Federated Search

The prominent features of federated search architecture are as follows:

- The architecture should provide configurable interfaces to add any number of internal and external information retrieval systems.
- The architecture should provide pluggable adaptors to various information sources to aggregate the information. The architecture should provide extensible integration architecture to accommodate custom plugins for future needs.
- The architecture should be able to aggregate information from systems, technologies, and products built on various platforms. The most preferred form of integration is through services.
- The architecture should enable the reusability of existing information systems by leveraging their search systems and information retrieval systems.

Sample Federated Search Architecture

We can implement federated search in multiple ways. Here we discuss two popular designs for federated search.

Federated search through results consolidation

In this case the federated search system submits a search query to various information retrieval systems. Each of those information retrieval systems executes search queries in parallel and provides search results from its sources. A federated search system then merges and sorts the search results.

Federated search through unified index

In this design, a federated search engine aggregates the information from various search indexes to create a combined and unified master index. A variation of this technique is to create various logical search collections for information sources and then combine the results from all search collections. Another variation of this technique is to have a virtual index pointing to various search indexes; a federated search engine submits the query to a virtual index that then gets the search results from associated search indexes.

Results consolidation and unified master index options are depicted in Figure 14.1.

Common Challenges with Federated Search

In the regular search design, we can process the crawled information at the document-processing stage. We can create a structured index that can be used in relevancy matching and results ranking. However, in case of federated search, the obtained search results pose few challenges:

Nonuniform results structure and result normalization: As we are working with disparate information systems, the search results obtained may be in different formats. As there is no common structure, it would be difficult to normalize the results. Hence, the first step in a federated search process is to bring all the search results into a common structure. Most federated search systems expect the search results from individual systems in XML format; otherwise, such a system would convert individual search results into XML format before merging them.

Information duplication: Search results obtained from multiple sources may contain duplicates. Those must be identified and removed. To achieve this, a federated search system may use the result identifier (name, path, ID, or URL) to eliminate duplicates. We can also cleanse the data wherever possible to remove duplicates. We discuss one such technique in the next section.

Overall relevancy ranking: As each search engine employs a different relevancy-ranking algorithm, it would be challenging for a federated search engine to accurately rank the aggregated results with uniform/consistent relevancy. A federated search engine can combine the aggregated search results using the relevancy specified by its source system. Another technique is to search within search results (as second pass) to determine their relevancy.

Security: Each information system contains its own security policies. First, federated search engines should connect to those systems with appropriate credentials.

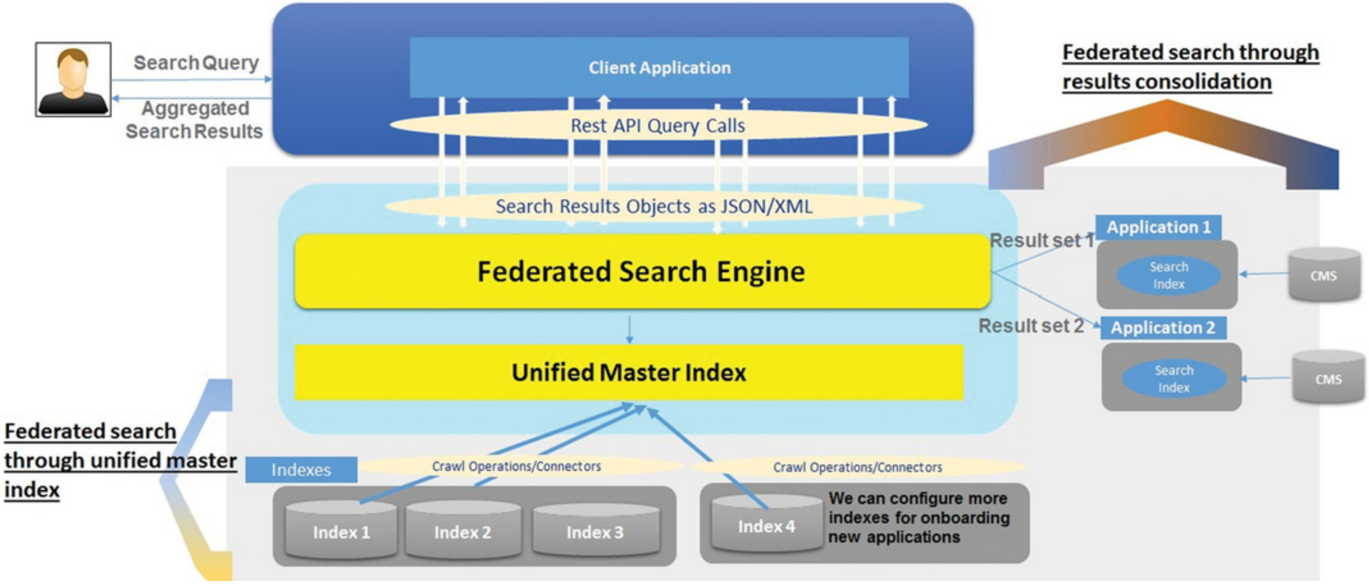

Search Query
Aggregated Search Results
Client Application
Rest API Query Calls
Search Results Objects as JSON/XML
Federated Search Engine
Unified Master Index
Federated search through results consolidation
Result set 1
Result set 2
Application 1
Application 2
Search Index
Search Index
CMS
CMS
Federated search through unified master index
Indexes
Crawl Operations/Connectors
Crawl Operations/Connectors
Index 1
Index 2
Index 3
Index 4
We can configure more indexes for onboarding new applications

Figure 14.1 Federated Search Architecture

A more challenging task is in terms of role interpretation. During secured search in a federated search scenario, we need to ensure that each of the federated systems interprets the user roles in a similar way. To address this issue, federated search can employ a federated security design using SAML-based assertions or use a uniform role structure or perform role translation before sending role parameters to the federated systems. Additionally, multilingual results and results encoded in various formats may pose challenges during results aggregation.

Enterprise Search through Intermediate Aggregation Repository (Alternative to Federated Search)

In an enterprise scenario consisting of various systems with a mix of structured and unstructured data, finding relevant content is difficult. Some of the challenges posed by such content are:

- Potential duplicate content due to replicated content among various systems
- Incompatible or unstructured data formats making it difficult to organize and structure index entries
- Data without any tag or metadata information making it difficult to find relevant content
- Multiple systems without any proper interface for search engines posing challenges in crawling and indexing content
- Various file types and document formats including the proprietary file format making it difficult to extract information
- Absence of a uniform classification scheme making it difficult to categorize the data.

As we have seen in the earlier sections, one of the ways to address this challenge is to use an enterprise-wide federated search for which we need an information retrieval service or interface for each of the enterprise data sources. However, if the enterprise systems do not have any existing information retrieval service or search interface, it may be difficult to use a federated search system to make sense of enterprise-wide data. In such scenarios, one of the most effective techniques is to aggregate the data of interest from the enterprise systems to an intermediate search-engine-friendly repository. A sample data integration process is shown in Figure 14.2.

We can leverage ETL (Extract, Transform and Load) tools to aggregate the data from multiple enterprise sources into an intermediate repository. During aggregation, ETL also transforms the data into a uniform structured format. Once the data is aggregated, it undergoes a “data cleansing” process that includes:

- Removing the duplicates of content or data
- Adding tags and metadata keywords to provide semantic meaning for content and help in identification of relevant content
- Classifying content into relevant logical categories

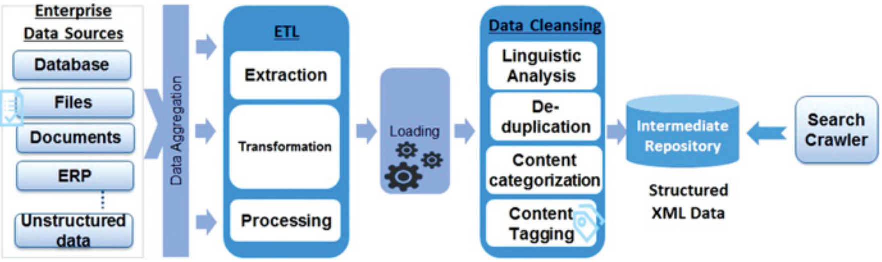


Figure 14.2 Data Integration into Intermediate Repository

Intermediate repository can hold processed content a in structured format, along with its metadata.

The primary goal of data aggregation and data-cleansing process is to improve the data quality for search engines. For the data-cleansing process to work effectively, we need to have a well-defined and controlled centralized vocabulary and metadata management system. This provides taxonomy of terms that can be used for content tagging and identification. Once the content loading and data cleansing are complete, an enterprise search engine can crawl the intermediate repository to index content.

14.2 ADVANCED SEARCH FEATURES

In this section we discuss some of the advanced search features such as relevancy adjustment, as well as alternative search suggestions and secured search.

Relevancy Rank Adjustment and Rank Boosting

In enterprise scenarios, a business wants to promote content and product that directly impacts business revenue. This is required to market newly released products or to announce important content updates. As search is one of the key user engagement and information discovery platforms, business wants to “promote” content during the search process. This can be achieved by “artificially” boosting the rank of promotional content so that it appears at the top of the organic search results list. Another way to achieve this is to provide a separate section (such as featured section or searchandizing section) to feature this content at the top of the search results list.

Most of the enterprise search engines provide ways to artificially boost the relevancy rank based on some criteria. Business administrators can adjust relevancy criteria and map content with relevant keywords to boost the rank. Relevancy rank can be adjusted using some of the following criteria:

- **Adjust rank for content of business interest:** In such cases, a business administrator can map the exact content URL to the query keywords.

- **Metadata-based ranking:** Business administrators can specify the metadata and map it to the keywords. All content tagged with such metadata will get higher ranking for the specific keywords.
- **Sorting criteria configuration:** Business administrators can specify default sort criteria such as content freshness, access frequency, or content rating. This will be used for default ranking of search results.

Additionally business administrator can provide the business term and synonyms for the keywords and map them to featured content to boost its relevancy rank. A search engine would then use this mapping to increase the ranking of the featured content/products in search results for a given set of keywords.

Apache Solr supports index-time boosting (boosting entire documents or fields during indexing) and query-time boosting (field-based boosting during query). Below is the query-time boosting for the *name* field:

```
(name:test)^2.0 (description:sample)
```

Personalized Search

Personalized search aims to enhance the relevance of search results for individual users. Personalization would happen at various levels and involves various factors. Ranking of search results may be varied based on user's context. Most commonly used attributes for personalized search are user's search history, user's attributes (roles, preferences, language), user's social connections, and user's context (access device, location, timestamp). User's context would add implicit personalization factors. For instance, a mobile search may add an additional filter to user's location; authenticated user search may add user roles as additional filters.

In a quest to make the search results relevant through personalized search, analytics plays an important role. Analytics tracks the user's response to search results and reports factors such as:

- Content format frequently employed by user (text, blog, map, video, image, etc.)
- Number of pages navigated
- Exit rate
- Usage of facets
- Preferred sort criteria
- Usage of synonyms, search suggestions

All these factors are "remembered" by the search engine and used for personalizing search results during subsequent search attempts.

In an enterprise scenario, users can explicitly specify their preferences and interests (such as content type, preferred language, default sorting criteria, etc.) These parameters, along with user's security attributes (such as roles, permissions), are used to personalize the search experience.

Alternative Search Suggestion

Sometimes users might not use most the appropriate search term to find what they are looking for. This will eventually impact the quality of search results. For instance, users might search for "*product debugging tips*" hoping to find good troubleshooting steps; a better search term would be "*product troubleshooting guide*," which would fetch more meaningful search results. Using synonyms is one of the natural ways to address this scenario. We can use all dictionary-based synonyms (after correcting the spelling of the original search term) and include the search results for the original term along with its synonyms in the search results.

Synonyms may not fully address all scenarios, however, because each business domain has its own vocabulary of business terms. For instance, "notebook" and "notepad" are related terms in book stationary business, whereas "notebook" and "laptop" are related terms in electronic retail. Hence, business administrators should define the related terms and synonyms in an enterprise taxonomy.

At runtime, a search engine will use the synonyms (both from a dictionary and from an enterprise taxonomy) to include results based on synonyms in search results. A search engine can use the "*related terms*" from the enterprise taxonomy to provide search suggestions such as "Do you want to search <related term>?" Analytics can also be used to find the relation between terms by tracking user searches. This can be used to provide alternate search suggestions such as:

"Users who searched for product A also searched for product B" where product A is the original search term and product B is the related term we have inferred from Web analytics.

Secured Search

A broken search process poses the risk of accidental information disclosure, providing private and confidential information in search results to an unintended audience. A search engine should be able to leverage the content security metadata/tags to enforce specified security policies and entitlement rules. A secured search process should include following aspects:

Transport-level security: The search page and the search results page should be delivered over secured transport channels such as HTTPS. This should be strictly enforced if the search collection includes confidential business content or private user data.

Document and asset access control list (ACL): Documents and digital assets will have an access control list (ACL) that specifies the users and roles along with their access permissions to content. A search engine should be able to identify the

ACLs during indexing and map them to the internal security tags that get added to the index structure.

The following are general secured search steps that use document and asset ACLs:

- **User authentication:** A user needs to be authenticated to access the secured search function. The Web application (with search function) authenticates the user from LDAP. After successful authentication, the application accesses user details such as roles, groups, and the like. Only valid users should be allowed to view the search page and the search results page.
- **Security filters with search:** When the user performs a keyword-based search, user's security attributes will be added as implicit filters to the search query along with the keyword. Implicit security filters include the following: search type (public/private), user roles, user groups, and user profile attributes. A search engine uses security filters to fetch only the authorized documents and content.
- **Document processing and indexing:** When indexing the document with ACLs, the search engine maps the document ACL to the internal search security tags. These search security tags will then be tagged to the content/document/asset as a security attribute. A sample mapping of a document ACL to a security tag is presented in Table 14.1.
- **Search query processing:** During query processing, a search engine will use the security filters provided by the search interface to match with security tags and filter the documents that satisfy the security criteria.

Content-level security: Security at content chunk level can be implemented using security metadata. For each of the content chunks, authors and content administrators can tag the security metadata such as target audience, user roles, user groups, and the like. Search engine maps it to internal security tags while indexing content and uses it for serving authorized content during content search.

Role-based access (RBA): RBA provides fine-grained access to content, data, and documents. Post-user authentication, user roles and groups are used to show only authorized content. The common way to achieve this is by appending the user roles

Table 14.1 Document ACL to search security mapping

Document ACL	Security Tag in Search
Visibility = public	Target_audience=public
ACL = Role 1 (full control), Role 2 (read permissions only), Role 3 (write permission only)	Target_audience=private and User_role = Role 1, Role 2 [Only roles with read permission are mapped to user roles to view the document]

Note: For a consistent role mapping from a document ACL to a search security tag, the document/asset repository and the search system should be connected to the same corporate user registry (LDAP) so that it is easy to have consistent and uniform role definition.

as additional filters in the search query along with the search keyword. We have seen the detailed flow while discussing document and asset ACL.

Data masking and encryption: The search crawler indexes confidential information such as Social Security number, credit card number, date of birth, and the like. While displaying such sensitive information in the search results page, there is a possibility of accidental information disclosure. In order to address this, the sensitive information should be masked. There are two ways to achieve this:

- **Mask the confidential information at the source:** Content systems or databases that are crawled by the search engine should mask the information.
- **Mask the confidential information at the intermediate repository:** We can create an intermediate data store that stores content from all sources (discussed as part of the "Federated Search" section). This intermediate store can mask the sensitive information. The search crawler can crawl and index this information.

Alternatively, a search engine can mask the data during content processing and entity extraction. However, it is recommended to mask the data at the source. This ensures that data is protected even when a user chooses to follow the link from the search interface (to reach the original document or content).

Secured search services and APIs: When the search engine exposes its function through Web services or APIs, the interface should enforce user authentication. In addition to this, the services should be exposed over secure transport layers such as HTTPS.

Data-level security: To implement data-level security, the database records should contain a column to indicate the role attribute. The search engine can then map the role to its internal security tag while indexing the data. This helps in implementing fine-grained data-level security at runtime.

Security for unstructured content: While indexing unstructured content such as e-mail, chat, blog, and feeds, the source system should provide a security identifier (such as authorized role, user, etc.) through security tags. The search engine can use it for mapping it to its internal security tag.

Security integration

For full enforcement of security policies, we need to consider the following aspects with respect to search engine:

- The access control list (ACL) should be clearly defined at the content level. The document or content should clearly indicate the role and access permissions. ACLs can be specified through special security tags or security metadata.
- The search engine should be capable of seamlessly integrating with a wide variety of user directories to get the user attributes and role information.

- The search engine has to take care of enforcing the entitlement rules by matching the content ACLs with the roles and permissions of the requested user. This filtering is required to provide role-based access.
- For crawling secured content or private content, the search engine should be configured with appropriate security settings such as access credentials, encryption standards, etc.
- While providing personalized search results, the search engine should also consider the security policies and user entitlements.
- Search results consisting of private and confidential personal information should be served only over HTTPS or similar secured channels.
- To prevent accidental information disclosure, a separate search collection for secured content can be configured. Access to this collection can then be restricted to enforce strict security policies.

A sample security module design for enterprise search is shown in Figure 14.3.

The diagram in the figure depicts a security module along with an enterprise search platform to enforce security policies. An authentication handler authenticates the user using a directory server. An ACL monitor ensures that the document-processing step adds the document ACLs in an appropriate format to the search index. The search filter generator adds implicit search filters (such as user roles, permissions) to the search query. The access rights module filters the results during the results-processing step.

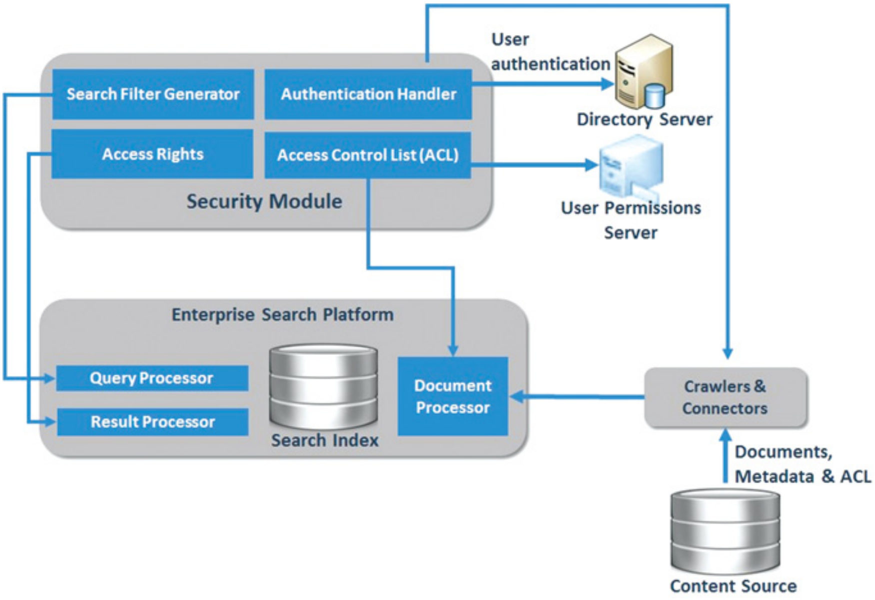


Figure 14.3 Search Security Module Design

14.3 ENTERPRISE SEMANTIC SEARCH

Semantic search is a popular recent trend among search engines. *Semantic search aims to provide relevant search results based on the "meaning" of the search terms and the "intent" of the user.* It goes beyond keyword-based search to understand the connection between various keywords for a given context. It relies on natural language processing, text analysis, and machine learning techniques to interpret a query term.

Semantic search requires semantically structured knowledge representation. A semantic knowledge structure can be easily read and interpreted by machines. One of the popular semantic structures is data represented in Resource Description Framework (RDF) language (consisting of subject-predicate-object triplet). "Meaning" is derived from the semantic structures using ontologies (a structured vocabulary that describes the concepts for a given domain). Some key differences between regular keyword-based search and semantic search are presented in Table 14.2.

In an enterprise context, semantic search is essential for handling informational and transactional queries. Enterprise semantic search would greatly benefit these domains:

- **Customer support:** Can help customers use natural language queries to get the most relevant information. For example, the enterprise semantic search

Table 14.2 Keyword search vs. semantic search

	Keyword-based Search	Semantic Search
Core search mechanism	Term matching, density and proximity of keywords	Search for semantics (meaning, context, relationships) of the user query, natural language processing, question-answer kind of search
Underlying techniques	Relevancy calculation, page ranking, inbound links calculation	Natural language processing, text analysis, concept extraction, relationship identification, pattern analysis, machine learning, fuzzy logic
Utility	Discovery of relevant information	Natural language query processing, personalized and contextual results, faceted search based on topic categories
Standards	XML, JSON	RDF, SPARQL, OWL
Relevancy calculation	Usually keyword position, proximity of keywords, keyword distribution, inbound links and their value	Meaning of the term, relevance to context

engine can answer customer queries such as "What is the latest driver for the product?" and "How do I troubleshoot performance issue for this problem?"

- **Retail and e-commerce:** Enhance transaction experience of customers and offer product-bundling choices. For example, the enterprise semantic search engine can answer user queries such as "Compare features of product A with product B", "What are the top selling products in this category?", "What are nearest stores from my location?"
- **Information management systems:** Generic information management portals that handle lot of structured and non-structured data from various enterprise content sources. Sample queries in this category are "Show me all the conversations in this topic."
- **Mobile search:** Semantic search engines facilitate natural language voice-based search in mobile and wearable systems.

Key Elements of Semantic Search

Taking a look at the main elements required for a semantic search engine helps us define the architecture and understand the search process.

Named Entity: Unambiguous representation of terms such as organizations, objects, place, and person are entities.

Taxonomy: Taxonomy is a structured organization of information. It represents the hierarchy of entities and classes of objects. We have discussed the details of taxonomy in Chapter 6. For example, a manufacturing business can maintain a robust taxonomy consisting of a product hierarchy.

Ontology: Ontology is a structured organization of concepts, concept properties, attributes, and relationships. Ontologies formally represent knowledge in a subject-object-predicate triple format that can be used for machine-based automatic processing and to infer knowledge. They represent knowledge about a domain in terms of concepts and vocabularies along with their relationship. Ontologies enable processing of semantic queries by providing a common structure for information and knowledge. They can semantically connect entities from various taxonomies and establish relationships across terms from different taxonomies.

Semantic search engines use ontologies to establish a relationship among entities and concepts and can create annotations for a specific domain. For instance, ontology can link the product name from a product taxonomy to a country in a geography taxonomy through a "can be sold in" relationship.

Natural language processing: The technique analyzes the text in natural language to understand its "meaning." It uses text analysis techniques to extract entities and concepts. Some semantic search engines also use machine-learning techniques for automatic content classification and automatic ontology term tagging. Resource description framework (RDF)-based data model and SPARQL are also used in some implementations of semantic search.

Enterprise Semantic Search Architecture

Semantic search architecture needs tools and techniques to interpret the natural language and establish relationships between extracted concepts. Semantic search architecture consists of enterprise ontology, search crawler, semantic search index, and semantic query-processing components.

Semantic search index

A semantic search index stores all the extracted concepts along with their synonyms, base forms, and their relationships with other entities. It provides a structured reference to semantically process a query.

Enterprise ontology

Ontology stores the business domain knowledge in a structured format. It consists of relationships between various entities. They can be used to annotate the documents and content for semantic analysis and categorization.

Semantic search crawler

When the search crawler crawls content and documents, the parser extracts the entities and concepts based on the identified language. Entity extraction pulls data related to organization, people, and place and then it establishes relationships with other enterprise data.

Semantic query processor

The component analyzes user's search terms using enterprise ontology.

Enterprise Semantic Search Process

Here are the general steps in the semantic search process (also depicted in Figure 14.4).

Crawling and indexing: A semantic search crawler crawls content sources. In addition to regular Web content and documents, a semantic search crawler can also crawl SKOS or similar semantic structures (W3C knowledge representation standards such as RDF, RDFS, OWL, etc.). The process uses text analysis and natural language processing, text mining, sentiment analysis, pattern matching, machine learning, and other similar techniques. Extracted concepts, named entities, and relationships are annotated/tagged using enterprise ontology (for instance, using rdfs:label to specify a resource name in OWL-based ontology); a tag can be triple, graph, category, etc. Content and documents are categorized based on taxonomy. Then it adds the extracted concepts and entities to the search index. In the process

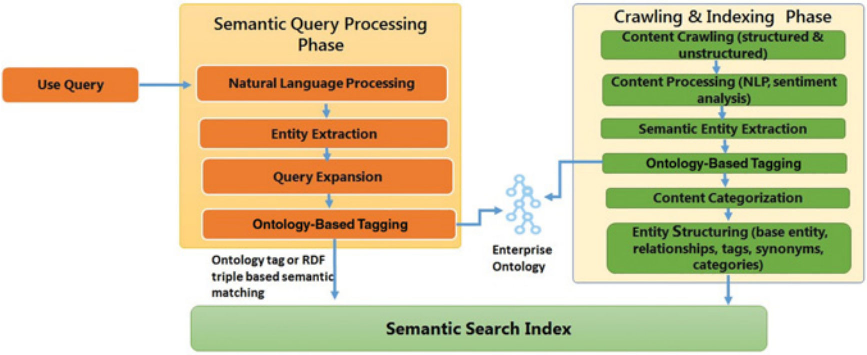

Semantic Query Processing Phase
Use Query
Natural Language Processing
Entity Extraction
Query Expansion
Ontology-Based Tagging
Ontology tag or RDF triple based semantic matching
Enterprise Ontology
Crawling & Indexing Phase
Content Crawling (structured & unstructured)
Content Processing (NLP, sentiment analysis)
Semantic Entity Extraction
Ontology-Based Tagging
Content Categorization
Entity Structuring (base entity, relationships, tags, synonyms, categories)
Semantic Search Index

Figure 14.4 Semantic Search Process

it also stores the base form of the entities, entity synonyms, entity categories, and relationship between extracted entities.

Query processing: During this phase, the query-processing component extracts the entities from the search term. Using natural language-processing techniques, a query processor expands the search term by adding semantically relevant terms. Entities are then mapped and tagged using enterprise ontology. Relevancy is calculated by using the "meaning" of search terms (using tagged ontology) with the "meaning" of indexed content. If the semantic crawler has extracted information in the form of RDF triple (subject-object-predicate), then an SPARQL query on RDF data can be used to get relevant matches.

Semantic Search Capabilities in Apache Solr

Apache unstructured information management applications (UIMA) framework can be used along with Apache Solr for adding natural language-processing capabilities. UIMA is designed to process unstructured content such as blog, e-mail, text, logs, etc. UIMA provides analysis capabilities such as language detection, tokenization, named entity annotation, and the like. Using UIMA, natural language-processing capabilities can be added to Solr at indexing time (using UpdateProcessor plugin) and during query-processing time (by customizing QParser plugin). Using UIMA we can do semantic faceting and search on named entities in unstructured content. The Solr UIMA plugin extracts named entities and knowledge during indexing.

14.4 PEOPLE SEARCH AND SOCIAL SEARCH

Collaboration needs in an enterprise scenario call for efficient people search. In an enterprise scenario, people search is mainly used for collaboration, resource staffing and allocation, and consultation. An enterprise search engine should index user information from various enterprise sources such as user repositories, project documents, source control systems, blog posts, wiki articles, skills database, HR ERP systems, and chats. Though it is important to index various attributes associated with an individual user, most commonly used fields in people search are first name, full name, skills, expertize, title/designation, and groups.

Challenges in People Search

Varied content sources: People's names can be sourced from a variety of structured and unstructured sources. While user repositories offer highly structured name formats, chats/articles are on other end of spectrum – they are often difficult to interpret.

Ambiguous name formats: As persons' names can be specified in various styles (first name, first name with last name, initials, with/without titles, truncated names, nicknames, alternative names, initials only, aliases, through titles and designations, localized names, etc.) in various systems, this would pose challenges for a search engine to establish a unique identity for a given person.

Cultural variations: Naming conventions such as spellings, naming formats, abbreviations, name changes post-marriage, and phonetic spellings vary based on cultures. In short, there is no universal naming structure that is consistent across all contexts.

People Search Design

Most enterprise search platforms provide a people search option. Generally, the people search functionality in enterprise search platforms involves these steps:

- Index name values (first name, last name, middle name, full name) in a search index as separate fields. We can get the structured name values from a user registry such as LDAP or HRMS.
- We can tag other related information against each name as a metadata value. Key useful tags are:
    - User's skills or expertize, obtained from a skill database or extracted from office documents and/or published papers
    - Title and designation obtained from an HR database or an ERP system
    - User's department and group information obtained from an HR database or an ERP system
    - User's interests, hobbies, activities, preferences, and other personal attributes, obtained by crawling personal pages and social pages.
    - User's article ratings, retrieved from various collaboration platforms
- During people search, the search engine can perform a field match along with tagged metadata using fuzzy logic.

Sample People Search in Apache Solr

Let us look at some of the Apache Solr features that can be leveraged for implementing people search.

Note: The steps given below are for educational purposes only. Based on the specific needs, filters and analyzers may vary.

- As a first step, index the full name as one of the fields. Optionally separate fields for the first name and last name can be created.
- Leverage eDisMax query parser for multi-column searching.
- Create a Solr-compatible synonym file (or use one of the available open-source Solr synonym files) to specify the aliases, nicknames, and other such name variations. Then use a "synonyms" token filter. For example, the following line in the synonym file will take care of all nickname variations of "Franklin"

```
Franklin, Franklyn, Frank, Frankie, Franky
```

- Use an analyzer with Lowercase and ASCIIFoldingFilter to convert all special characters in names to ASCII characters.
- Perform people search using fuzzy logic.

Social Search

Social search involves searching and analyzing information from a user's social graph. It includes recommendations driven by social networks, people search, search within social circles, user's business connections, social suggestions, and the like. The user's social factors form the search context within which search happens. Social factors are used as social filters during search.

In this category the search engine processes and indexes content coming from unstructured sources such as blogs, wiki, social networking sites, forums, communities, etc. Most of this content is user generated (UGC). Due to rapid explosion of UGC, social search is closely associated with the big data challenge.

Generally, the user's social factors (such as social connections, social activities, social network groups) signal the relevancy stronger than traditional relevancy factors. The core features of social search are:

- Enterprise social conversations (e-mail, chats, blogs, comments, reviews, enterprise collaboration platforms) of the user's social connections form part of the search results.
- Content generated by the user's social connections (such as posts, answers, comments) will be included in social search.
- Trending topics, hot topics, and other current content may influence the social search results.

- Relevancy is influenced by social likes, shares, tweets, +1 ratings, number of followers, and similar social signals.
- Content authority is influenced by author authority (using data such as number of followers), peer feedback about the content and author, and other similar factors.

The main technologies involved in social search are big data search, sentiment analysis, and natural language processing.

14.5 MOBILE SEARCH

Search is increasingly used as a primary tool in a mobile experience. A direct application of search technology in mobile space is *voice search.* A mobile user speaks out the search query and the search engine has to perform natural language processing to find the relevant results. Other salient features of mobile search are:

- **Mobile context**: Mobile search gets additional context parameters compared to regular search. Main context parameters are geo-location, device details, timestamp, user's preference, and user's search history. Search relevancy needs to consider these context parameters to find the most relevant results.
- **Actionable results:** During search results presentation, it is important to provide most relevant actionable results instead of the regular results summary. Due to limited real estate on a mobile device, mobile users look for "actionable answers" instead of a list of results.
- **Virtual digital assistants:** Most of modern smart phones provide digital assistants who interact with the mobile user through voice. They take care of many day-to-day tasks such as meeting scheduling, reminders, information search, point-of-interest search, etc. Such software relies heavily on semantic search and natural language processing search.
- **Search-based applications:** There are many search-based application for mobile platforms. Software such as music search, maps, social search, voice search, and location-based search all use search as the backbone.

14.6 BIG DATA SEARCH

Big data stands for a large volume of data (usually greater than 1 TB) that an organization cannot effectively manage using traditional data management tools and techniques. The main characteristics of big data are volume (data volume), velocity (rate of data change), and variety (types of data formats). Over a period of time, enterprises accumulate excessively large amounts of data in structured, semi-structured, and unstructured formats. Social and collaboration platforms produce high-volume user-generated content (UGC) at a fast rate. Other channels such as chat, blogs, e-mail, and reports also contribute to unstructured data volume.

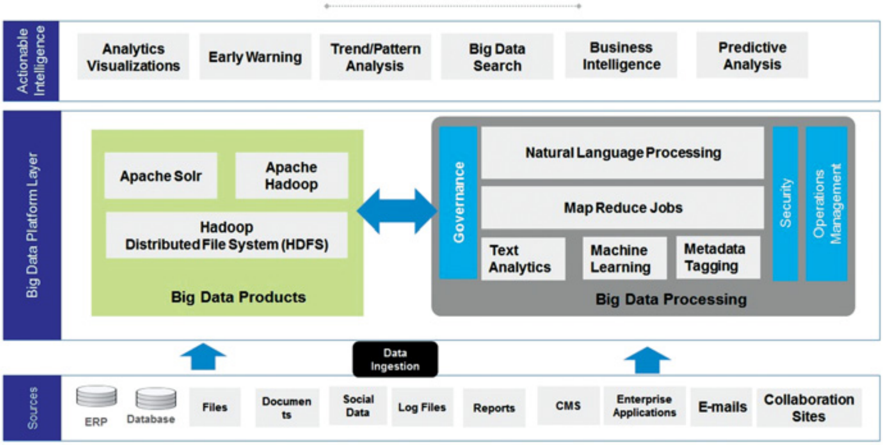


Figure 14.5 Enterprise Big Data Setup

Searching and processing big data provides many actionable insights to enterprises. It helps enterprises understand users' sentiment, identify trends and patterns, help in building analytics visualizations, and provide an early warning mechanism. A big data setup in an enterprise scenario is depicted in Figure 14.5.

The information sources consist of rapidly growing enterprise data. It includes structured data sources such as database, files, ERP data, and CMS content along with unstructured data such as blogs, chat, e-mails, collaboration content, social data, logs, and the like. A big data platform ingests the data for further processing. Such a platform consists mainly of technologies to process big data. Natural language processing, text analytics, and machine learning are some of the widely used technologies in big data processing. Apache Hadoop, for example, uses the MapReduce technique (which involves partitioning large problems into small maps and solving each map simultaneously on cluster nodes) and Hadoop Distributed File System (HDFS). The processed information can be used to obtain actionable intelligence. We can create analytical visualization, early warning systems, analyze emerging trends and patterns, perform big data search, do a predictive analysis, or obtain business intelligence reports, among other activities.

Note: The technologies indicated in the Figure 14.5 are solely for educational purposes. We can use other alternate big data technologies to build a similar platform.

Let us now look into usage and integration of open-source technologies for big data search.

Apache Solr Integration with Apache Hadoop

Apache Hadoop is one of the popular big data platform that handles big data search and processing. It also uses Apache Solr for text searching. Apache Solr can also be used to index the processed big data directly. One such setup is shown in Figure 14.6.

Sources such as logs, social conversations, chat, sensor data, and collaboration constitute big data and are ingested into the big data platform. In this scenario, the big data platform uses map reduce to process the data and stores the processed data in a clustered index. Apache Solr can crawl the clustered index directly and use the processed big data in search results.

Apache Shards for Searching a Large Dataset

Apache Solr also scales up well for a large dataset. We can leverage shards (part of a distributed index, consisting of elements such as disjointed replicated data from a collection) and SolrCloud for searching large-sized data.

14.7 SEARCH ENGINE OPTIMIZATION (SEO)

Search engine optimization (SEO) is concerned with increasing the visibility of Web channels by making them more search engine friendly. By following SEO best practices we can make it easier for a search engine to crawl and index content. This also ensures that Web site pages and content get the right relevancy and appropriate ranking in search results. We have seen the SEO guidelines from content standpoint in chapter 10.

Besides visibility and relevancy, SEO has other benefits:

- SEO adds value to digital marketing and branding efforts. If digital campaigns and site content appear at the top of a search results list, they are more likely to be used by end-users.
- SEO improves the effectiveness of the messaging strategy by placing related content in the right place at the right time.
- SEO positively influences information discovery.

The following sections take a look at some of SEO optimization techniques.

Page-Level SEO Tags

Figure 14.7 depicts various page-level SEO-friendly measures.

Each Web page should contain meaningful page-level keywords and meta-tags.

```
<meta name="description" content="The page contains list of electronic items">
<meta name="keywords" content="Television, display screens, electronic">
```

Meta-tags can contain other meaningful attributes such as language, character set, updated date, author, and others. Search engines can use this metadata for determining the relevancy of page content. The page title is another important attribute of SEO. The title should be descriptive and should preferably describe the fundamental content.

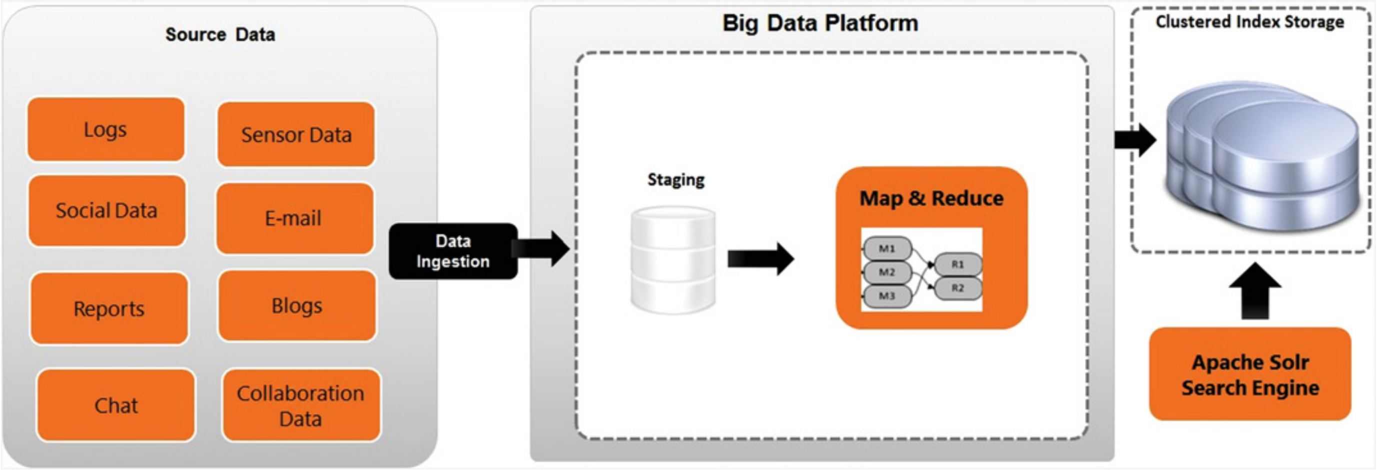


Figure 14.6 Apache Solr for Big Data Processing

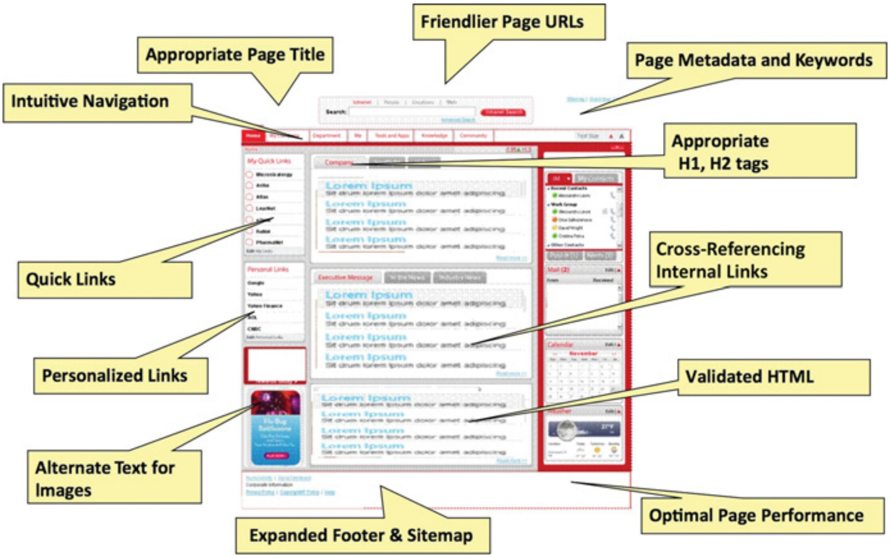


Figure 14.7 Page-Level SEO Measures

A page title with a matching search term is given higher priority when determining relevance. Site structure should be indicated through sitemaps and other navigation elements; crawlers should be able to reach all pages through the links and navigation elements. Quick links and personalized links may help search engines reach important pages. All binary and non-text content should have a text alternative (such as alt text for images). A Web page containing valid HTML and with good performance is SEO friendly.

SEO Strategy

Four key elements of SEO are depicted in Figure 14.8.

Content SEO: This involves tagging content with meaningful and semantic tags. For each content chunk we can specify keywords and metadata attributes. Specified keywords should be relevant to content; keyword density and proximity also play a key role in SEO strategy. We have discussed content SEO strategy in detail in Chapter 10.

Web Page SEO: At page level, we need to have a meaningful page title and structured URL. Page title and URL should preferably contain main keywords; a thorough research needs to be done to understand most appropriate keywords. Wherever possible, SEO blockers such as flash files should be avoided or we textual alternatives for those objects should be provided. Sitemaps and robots files may help search engines understand the site structure and crawl the allowed URLs. Quality inbound links need to be provided by linking main pages from external blogs, affiliate sites, feeds, wikis, social networking sites, and campaign pages. Once the pages are ready, we can submit the URLs to search engines for indexing.

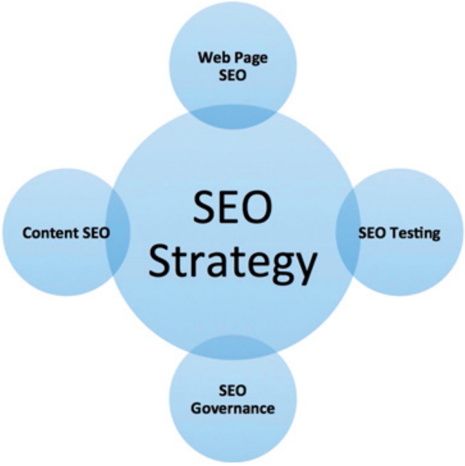


Figure 14.8 SEO Strategy

SEO Testing: Use the guidelines provided by popular search engines (such as Google Webmaster guidelines, Yahoo Webmaster guidelines, Bing Webmaster guidelines) to ensure that the Web pages follow SEO best practices and recommended guidelines. We can also involve SEO analyzers and checkers into testing the Web pages.

SEO Governance: Once the Web pages are live, we need to monitor the page rankings (for the specified keywords) in popular search engines. Continuously monitor other metrics such as site traffic rate, conversion rate, and traffic from search engines. Based on the success of SEO measures, we can fine-tune SEO parameters and enhance marketing activities. In order to improve visibility, we can adopt other SEO marketing techniques such as sponsored links (for specific keywords), social media marketing campaigns, and other search engine marketing (SEM) techniques. Other promotion and marketing activities are linking pages from external blogs, forum posts, articles, and social networking sites. SEO governance is also concerned with development of standards and processes for optimal SEO design, customization, and validation.

We need to tie the SEO efforts to the specific, ultimate business goals. We have discussed KPIs to measure business goals in Chapter 10 while discussing content KPIs. Metrics such as conversion rate, user registrations, site traffic, online revenue, subscriptions, and content sharing are indicative of success factors for business. We need to closely track these business goal indicator metrics to continuously fine-tune SEO efforts.

SEO Best Practices

Some of the effective SEO best practices from the enterprise search standpoint are as follows:

- Ensure that content is search engine friendly as much as practically possible. Provide valid text-based content and provide text-based alternatives for nontextual content so that search engines can easily index it.
- Use appropriate keywords and page-level metadata for pages. Specify only keywords relevant for content and which have high density in the current content.
- Adopt user-friendly page URLs. If there are dynamic URLs, internally translate them to static URLs.
- Develop an intuitive navigation model and site structure so that all pages on the site are reachable from the home page through links, menus, and header/footer. For non-linkable pages provide a sitemap file linking to all those pages. If page links are embedded in images/videos/flash files/PDF files, provide those links outside these nontextual entities so that a search engines can index them.
- For non-text binary content (such as images, videos, animations), always provide a text alternative (such as alt text for image, transcript for video, etc.)
- Provide sitemap (to specify site structure) and robots file for the site (to indicate the pages and site sections that should not be crawled).
- For external pages, submit the sitemap XML explicitly to search engines.
- Optimize the page performance.
- Externalize JS and CSS files.

SEO Anti-Patterns

The following are some of the common anti-patterns that we can avoid from an SEO standpoint:

- Avoid using iframes and flash content, which can pose challenges to search engines
- Avoid using cryptic and dynamic URLs; provide friendlier URLs that can be internally mapped to the dynamic URL
- Avoid duplicate content spread across multiple pages, as it can impact search ranking
- If there are dynamic pages, they should be reachable by search engines through a seed list
- Avoid using page redirects
- Avoid using irrelevant keywords or page titles
- Minimal social engagement and absence of social media strategy may impact SEO
- Absence of alt text and description for digital assets may impact visibility of binary content

- Minimize all page redirects and remove all 404 (resource not found) errors
- Avoid dynamic URLs
- Avoid spamming keywords for a given page

14.8 CASE STUDY: INFORMATION MANAGEMENT PORTAL DRIVEN BY APACHE SOLR

Background and Context

In this case study we look at building an information management portal for an enterprise. The main requirement for this portal is to index Web content and the documents from varied content sources and to serve relevant content to the user. The portal should provide an intuitive user interface that will actively engage employees and other internal business stakeholders.

Information Management Portal Solution Components

We can leverage an open-source portal product along with CMS and enterprise search to build the overall solution. The conceptual architecture is depicted in Figure 14.9.

The solution uses MVC architecture with layered separation of architecture concerns. The main features of each of the layers are explained below.

Portal Presentation

- Portal built-in capabilities of access management may be leveraged for access management.

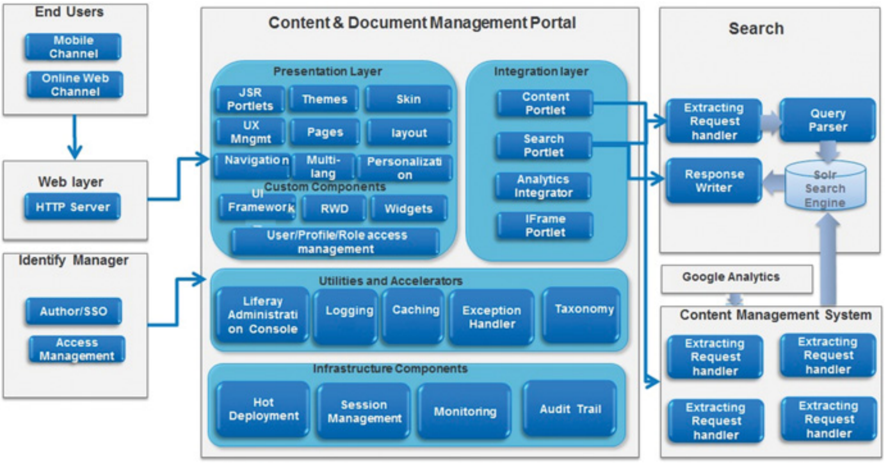


Figure 14.9 Content and Document Management Portal

- Portal presentation can be leveraged for presentation of content and customized to provide a rich user experience.

Content Management System

- A content management system can act as a content repository and hold the Web content information from all other systems.
- There can be a periodic scheduler, which will sync up content between the enterprise content systems and the content management system.
- Content management provides the following features to the solution:
  - Single unified content management system, from which the portal can fetch the information
  - The search engine can index the complete information and aid in efficient searching of any keyword in content
  - The solution is scalable, as we are using a portal that can support integration with enterprise systems, as well as any dynamic requirements, in the future.

Enterprise Search System

The crawling and indexing process is depicted in Figure 14.10.

- Apache Tika will be used for extraction of all the contents of the documents and pushing them to the Solr index. This also includes the metadata of the documents.

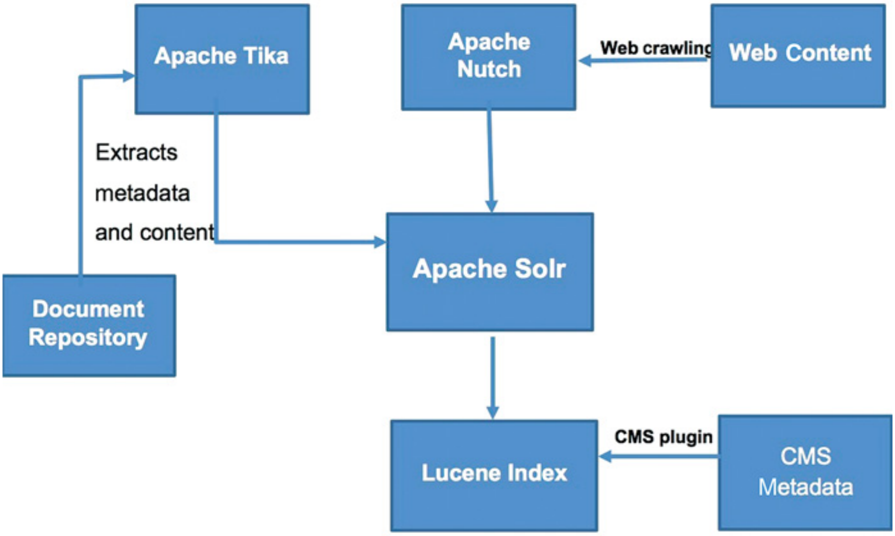


Figure 14.10 Search Crawling and Indexing Process

- Apache Nutch is a Web crawler and will be crawling all the systems and portal for all the contents in the CMS system.
- Apache Solr is primarily a full-text search engine that indexes all the data provided by Apache Tika and Nutch. This data is stored as a lucene index.
- The presentation portal is configured with Apache Solr to search for Web content and documents.

14.9 CHAPTER SUMMARY

- The chapter discussed advanced search concepts such as federated search, advanced search, semantic search, search KPIs, SEO, and search governance.
- Federate search distributes a search query to various information retrieval systems for parallel execution. Federated search is required when we need to aggregate search results from various systems with their own search systems.
- The two main designs of federated search are federated search through results consolidation and federated search through unified index.
- In federated search through results consolidation, we aggregate results from various information systems and merge them.
- In federated search through unified index, we aggregate the information from various search indexes to create a unified index.
- Challenges in results aggregation during federated search include results structure, information duplication, relevancy ranking, and security.
- Enterprise search through intermediate aggregation repository is an alternative to federated search wherein we use ETL and such tools to aggregate data from enterprise systems into an intermediate repository. Enterprise search engine would then index the intermediate repository.
- Search filters can be used along with query terms to filter the search results.
- Relevancy rank adjustment can be adjusted by adjusting rank for content of business interest, metadata-based ranking, and sorting criteria configuration.
- Alternative search suggestion can be implemented using synonym support and using relationships configured in taxonomy.
- Secured search can be implemented using transport-level security, document ACL, content-level security, data masking and encryption, secured search services and APIs, data-level security, and security for unstructured content.
- Semantic search aims to provide relevant search results based on the “meaning” of the search terms and the “intent” of the user.
- Key elements of semantic search are named entity, taxonomy, ontology, and natural language processing.

- Semantic search engine architecture includes semantic search index, enterprise ontology, semantic search crawler, and semantic query processor.
- In semantic search flow, in the crawling phase, the extracted name entities are annotated/tagged with ontology terms. In the query-processing phase, relevancy is calculated by using the “meaning” of search terms (using tagged ontology) with the “meaning” of indexed content.
- UIMA plugin enables NLP capabilities in Apache Solr search engine.
- Main technical search KPIs are search exit rate, search performance, zero-result query, query metrics, and successful search rate.
- Main business search KPIs are customer satisfaction index, conversion rate, most popular keywords, percent reduction in support calls, and percent improvement in issue resolution time.
- SEO strategy includes content SEO, Web page SEO, SEO governance, and SEO validation.
- Search governance includes search administration process, on-boarding process of a new information source, search tracking and monitoring, search maintenance and operations, and process for handling emergency updates.

Further Reading

Boiko, Bob (2004), *Content Management Bible*, 2nd Edition, Wiley.

Casey, Meghan (2015), *The Content Strategy Toolkit: Methods, Guidelines, and Templates for Getting Content Right (Voices That Matter)*, 1st Edition, New Riders.

Halvorson, Kristina and Rach, Melissa (2012), *Content Strategy for the Web*, 2nd Edition, New Riders.

Hearst, Marti A. (2009), *Search User Interfaces*, 1st edition. Cambridge University Press.

Kissane, Erin (2010), *The Elements of Content Strategy*, A Book Apart.

Morville, Peter and Callender, Jeffery (2010), *Search Patterns: Design for Discovery*, 1st Edition, O'Reilly Media.

Nichols, Kevin (2015), *Enterprise Content Strategy: A Project Guide*, XML Press.

Nichols, Kevin and Rockley, Ann (2015), *Enterprise Content Strategy: A Project Guide*, New Riders.

Nudelman, Greg and Gabriel-Petit, Pabini (editor) (2011), *Designing Search: UX Strategies for eCommerce Success*, 1st Edition, Wiley.

Rockley, Ann and Cooper, Charles (2012), *Managing Enterprise Content: A Unified Content Strategy (Voices That Matter)*, 2nd Edition, New Riders.

White, Martin (2012), *Enterprise Search*, 1st Edition, O'Reilly Media.

White, Martin (2015), *Enterprise Search: Enhancing Business Performance*, O'Reilly Media.

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent

Index

Enterprise Content and Search Management for Building Digital Platforms, First Edition. Shailesh Kumar Shivakumar.

Companion website: www.wiley.com/go/shivakumar/enterprisecontent