



LIBRARY AND INFORMATION SCIENCE TEXT SERIES

INTERNET TECHNOLOGIES AND INFORMATION SERVICES

SECOND EDITION



Joseph B. Miller

Internet Technologies and Information Services

Recent Titles in Library and Information Science Text Series

Basic Research Methods for Librarians, Fifth Edition
Lynn Silipigni Connaway and Ronald R. Powell

Public Libraries in the 21st Century
Ann E. Prentice

Introduction to Technical Services, Eighth Edition
G. Edward Evans, Jean Weihs, and Sheila S. Intner

Science and Technology Resources: A Guide for Information Professionals and Researchers
James E. Bobick and G. Lynn Berard

Reference and Information Services: An Introduction, Fourth Edition
Richard E. Bopp and Linda C. Smith, Editors

Collection Management Basics, Sixth Edition
G. Edward Evans and Margaret Zarnosky Saponaro

Library and Information Center Management, Eighth Edition
Barbara B. Moran, Robert D. Stueart, and Claudia J. Morner

Information Resources in the Humanities and Arts, Sixth Edition
*Anna H. Perrault and Elizabeth Aversa, with contributing authors
Cynthia Miller and Sonia Ramírez Wohlmuth*

The Collection Program in Schools: Concepts and Practices, Fifth Edition
Kay Bishop

The School Library Manager, Fifth Edition
Blanche Woolls, Ann C. Weeks, and Sharon Coatney

Young Adult Literature in Action: A Librarian's Guide, Second Edition
Rosemary Chance

Children's Literature in Action: An Educator's Guide, Second Edition
Sylvia M. Vardell

INTERNET TECHNOLOGIES AND INFORMATION SERVICES

Second Edition

Joseph B. Miller

Library and Information Science Text Series



LIBRARIES UNLIMITED

AN IMPRINT OF ABC-CLIO, LLC
Santa Barbara, California • Denver, Colorado • Oxford, England

Copyright 2014 by ABC-CLIO, LLC

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, except for the inclusion of brief quotations in a review, without prior permission in writing from the publisher.

Library of Congress Cataloging-in-Publication Data

Miller, Joseph B., 1952-

Internet technologies and information services / Joseph B.

Miller. — Second edition.

 pages cm. — (Library and information science text series)

 Includes bibliographical references and index.

 ISBN 978-1-61069-473-5 (pbk: alk. paper)

 1. Libraries and the Internet. 2. Libraries—Information technology. I. Title.

 Z674.75.I58M55 2014

 020.285'4678—dc23 2014013761

ISBN: 978-1-61069-473-5

18 17 16 15 14 1 2 3 4 5

Libraries Unlimited

An Imprint of ABC-CLIO, LLC

ABC-CLIO, LLC

130 Cremona Drive, P.O. Box 1911

Santa Barbara, California 93116-1911

This book is printed on acid-free paper 

Manufactured in the United States of America

To my wife Susan, for a lifetime of love and support

This page intentionally left blank

Contents

Preface.....	xvii
Acknowledgments	xxi

PART 1: Internet Technologies

1—Introduction	3
The Internet and the Changing IT World	3
The Internet Defined	5
A Brief History of the Internet.....	5
Administration of the Internet	7
<i>Net Neutrality</i>	8
IT Foundations.....	9
<i>Computers: The Binary Machine</i>	9
<i>Client–Server Architecture</i>	13
The Internet and Technology Trends	13
<i>Computing to Connecting</i>	13
<i>Bandwidth</i>	14
<i>Going Mobile in a Post-PC World</i>	15
<i>There's an App for That</i>	15
<i>The Cloud</i>	16
<i>Big Data</i>	16
<i>Going Open Source</i>	16
<i>Internet-Based Collaboration</i>	17
<i>The Long Tail</i>	17
<i>The Internet of “Everything”</i>	17
Summary	19

viii Contents

Notes.....	20
References.....	20
2—The World Wide Web.....	23
The Development of the Web.....	23
Key Web Technologies: URLs, HTTP, and HTML	25
The Invisible Web: Below the Surface.....	26
Web 2.0.....	27
The Mobile Web	27
The Social Web	28
<i>Blogs and RSS</i>	29
<i>Wikis</i>	29
<i>Social Networking Sites</i>	29
Virtualization, Grids, and Clouds	30
<i>Virtual Machines and Virtualization</i>	30
<i>Grids</i>	32
<i>Cloud Computing</i>	33
<i>Cloud Architecture</i>	34
<i>Cloud Application Examples</i>	35
<i>Extending the Browser</i>	36
Using the Web	37
<i>Information Seeking</i>	37
<i>eCommerce</i>	37
<i>The Web and Traditional Media</i>	38
<i>Education</i>	39
Summary	40
References.....	40
3—Network and Connection Technologies	43
Network Basics	43
The OSI Network Reference Model	44
Key Concepts and Terminology.....	45
Network Hardware	46
<i>The Network Interface</i>	46
<i>Packet Forwarding and Routing Hardware</i>	46
Wired Network Topologies.....	47
Wireless Networks	50
Protocols: Rules for a Common Language	51
Ethernet.....	52
Virtual Private Networks.....	54
Proxy Servers	54
Connecting to the Internet.....	55
<i>The Internet Service Provider (ISP)</i>	55
<i>Modems</i>	56
<i>Analog Phone Service: POTS and PSTN</i>	57
<i>Broadband Services</i>	57
<i>Leased Telephone Lines</i>	58
<i>Integrated Services Digital Network</i>	58
<i>Digital Subscriber Line</i>	59

<i>Cable Broadband</i>	59
<i>Satellite Internet Service</i>	60
<i>Fiber Options: FiOS Internet, U-Verse, and Google</i>	60
<i>Wi-Fi Broadband</i>	61
Mobile and Cellular Data Services	62
<i>Mobile Phones</i>	62
<i>Mobile Layers</i>	63
<i>Data Services</i>	64
Broadband Technologies and the Digital Divide	65
Summary	66
References.....	67
4—Internet Technologies: TCP/IP.....	69
Packet Switching and TCP/IP	69
IP Packet Address Headers	71
ARP	74
TCP, UDP, and ICMP	74
IP Addressing	76
<i>Private IP Addresses, NAT, and APIPA</i>	79
<i>IPv6</i>	80
<i>Managing IP Address Assignments</i>	80
Routing and Subnet Masks	83
Firewalls and Proxy Servers.....	87
The Domain Name System.....	87
<i>Domain Name Registration</i>	88
<i>DNS Lookups</i>	90
TCP/IP, System Layers, and OSI.....	91
Command Utilities for TCP/IP	92
Summary	93
References.....	93
Additional Reading	94
5—Higher-Level Internet Protocols: Making the Internet Work	95
Email: SMTP, POP, IMAP	96
Real-Time Internet Connections	98
<i>Telnet and TN3270</i>	99
<i>RTP and IRC</i>	99
File Management with FTP	100
Protocols for Information Services	100
<i>The Gopher Protocol</i>	101
<i>The Hypertext Transfer Protocol</i>	101
<i>HTTP and Statelessness</i>	105
<i>HTTP Secure</i>	106
<i>SNMP and LDAP</i>	107
Summary	107
References.....	108
Additional Reading	108

6—Internet-Connected Devices and Security.....109

Security Issues for PCs and Mobile Devices	111
TCP/IP and Ports	112
Internet Client Programs and Security.....	113
Internet Server Security	114
Threats and Issues.....	115
<i>Viruses, Worms, and Trojans</i>	115
<i>Rootkits</i>	117
<i>Spam and Phishing</i>	117
<i>Hoaxes</i>	120
<i>Fake Sites, Pharming, and Honeypots</i>	120
<i>Cookies and Web Bugs</i>	120
<i>Bots and Spyware</i>	121
<i>Wi-Fi Eavesdropping and Spoofing</i>	121
<i>Mobile Device Threats</i>	122
Anatomy of an Attack	122
Security Responses	124
<i>Antivirus Programs</i>	124
<i>Firewalls, Proxies, Routers, and VPN</i>	124
<i>OS and Office Application Updates</i>	125
<i>Password Security</i>	125
Warning Signs of a Problem.....	127
Security “Top-Ten List”	129
Summary	129
References.....	130

PART 2: Building the Web

7—Web Design and Graphics.....135

Web Design Overview	135
<i>Design Teams for Website Development</i>	136
<i>Site Mission and Audience</i>	138
<i>General Design Guidelines</i>	139
<i>Key Elements of Web Design</i>	140
<i>User Experience Design</i>	143
<i>Web 2.0 and Design</i>	143
<i>Mobile 2.0 and Design</i>	145
<i>Responsive Design</i>	146
<i>Tips for Mobile Designers</i>	146
<i>Apps vs. Website.</i>	147
Site Planning.....	147
<i>Schematics</i>	147
<i>Layout</i>	149
<i>Site Organization</i>	151
<i>Hypertext</i>	153
<i>Navigation</i>	153
Typography and Fonts.....	154
Graphics and Color in Design.....	155
<i>Analog to Digital</i>	156
<i>Color Use</i>	157

<i>Image Symbolism</i>	158
<i>Image Maps</i>	158
<i>Hardware and Graphics</i>	158
<i>Graphic Types</i>	159
<i>Color Schemes</i>	161
<i>Resolution Issues</i>	162
<i>Graphic File Formats</i>	163
<i>Using Images and Multimedia in Web Pages</i>	165
Accessibility	166
Usability and Findability	168
Design and the Lowest Common Denominator.....	168
Summary	169
References.....	170

8—Web Publishing with the Hypertext

Markup Language	173
Markup Languages.....	174
The Hypertext Markup Language.....	175
<i>The HTML 4.01 Specification</i>	176
Physical vs. Logical Markup	178
Elements, Tags, and Attributes.....	179
<i>Deprecated Tags and Attributes</i>	180
<i>Inline and Block-Level Elements</i>	180
Using HTML: Common Tags	181
<i>Images and HTML</i>	182
<i>The Hypertext Reference</i>	183
<i>Relative vs. Absolute References</i>	184
Tables	187
Forms.....	187
<i>Image Maps</i>	192
Framesets and Frames.....	193
<i>Inline Frames</i>	194
<i>Hypertext Links and Frames</i>	196
<i>The Base Tag</i>	196
XHTML.....	196
Mobile Markup Languages.....	197
WML.....	197
XHTML-MP	198
HTML5.....	198
<i>Key Differences: HTML5 vs. HTML 4.01</i>	199
<i>Major Advances with HTML5</i>	200
Summary	201
References.....	201
Additional Reading	202

9—Controlling Presentation with Styles **203**

The Box Model	205
CSS Positioning.....	206

Adding Styles to HTML	206
<i>External Style Files</i>	207
<i>Embedded Styles</i>	208
<i>Inline Styles</i>	208
CSS Syntax and Rules.....	208
CSS Rules	208
CSS Classes.....	209
<i>Pseudo-Classes</i>	211
<i>Sizing and Color Units in CSS</i>	211
Understanding the Cascade.....	213
HTML5 and CSS3 Modules.....	214
<i>CSS3 Modules</i>	214
CSS Hacks and Shortcuts	215
Responsive Design and CSS	216
Putting IT Together: A CSS Example.....	219
Summary	225
References.....	225
 10—Introduction to Web Programming.....	227
Concepts and Terminology	228
Web Programming	229
Databases and Web Programming	230
App Programming.....	231
Script Programming Basics	231
<i>Software for Script Programming</i>	233
<i>Client-Side vs. Server-Side Programming</i>	233
<i>Pros and Cons of Server vs. Client Execution</i>	235
<i>Programming Details for Scripts</i>	235
JavaScript Basics.....	237
<i>JavaScript Examples</i>	239
<i>JQuery and MooTools</i>	243
PHP Overview	243
<i>PHP Syntax</i>	244
<i>PHP: First Example</i>	245
<i>PHP Arrays and Functions</i>	246
<i>PHP: HTML Form Processing Example</i>	248
<i>Global and Superglobal Variables</i>	250
Summary	252
References.....	253
Additional Reading	253
 11—Web Technologies and Content Management	255
Content Management Systems	255
Databases and Dynamic Websites	256
<i>Relational Database Basics</i>	256
WCMs	260
CMS Examples	262
<i>Google Sites, Weebly, and Wix</i>	263

<i>WordPress</i>	263
<i>Drupal</i>	264
<i>LibGuides</i>	265
<i>Learning Management Systems</i>	266
Digital Repositories and Archives.....	266
<i>The OAIS Reference Model</i>	267
<i>Microservice Architecture</i>	269
<i>Digital Repository Examples</i>	270
Summary.....	271
References.....	272
12—XML Primer	275
XML: The Extensible Markup Language.....	277
XML vs. HTML.....	277
XML Key Concepts	278
<i>Key Components of the XML Specification</i>	279
<i>XML and Data Structures</i>	280
<i>Creating XML Documents</i>	281
<i>Well-Formedness and Validity</i>	282
The Document Type Definition (DTD).....	283
<i>DTD Elements</i>	284
XML Schema Definition	286
<i>XML Schema Definition Example</i>	287
<i>XML Schema Organization</i>	290
Vocabularies and Namespaces.....	292
Viewing and Processing XML	293
<i>CSS and XML</i>	294
<i>XSL</i>	296
XML-Related Technologies.....	296
XML Implementations	297
<i>RDF and RDFa</i>	298
XHTML.....	299
Alternatives to XML.....	300
<i>JSON</i>	300
Summary	302
References.....	303
Additional Reading	304
PART 3: Internet Content and Information Retrieval	
13—Internet Content	307
Content Challenges	308
File Basics.....	308
Standards and Formats.....	309
Presenting Content on the Web.....	310
Common Document Formats.....	311
<i>eBooks</i>	311

Collaborative Content.....	313
<i>Email Discussion Lists and Usenet</i>	313
<i>Blogs, News Feeds, Tweets, and Podcasts</i>	314
<i>Wikis</i>	315
Multimedia on the Web.....	315
<i>Creating Sound and Video Files</i>	315
<i>Streaming Audio and Video</i>	317
<i>Common Multimedia Formats</i>	318
Compression and Encoding Formats	320
Internet Resource Evaluation	320
Summary	322
References.....	323
 14—Information Retrieval.....	325
Information Retrieval Overview	325
Textual IR Systems.....	327
Indexes and Inverted Files	328
Vocabularies.....	330
<i>Controlled Vocabularies</i>	331
<i>Automatic Indexing</i>	332
<i>Query Formation</i>	332
Performance Measures	334
Classical IR Models	335
<i>Boolean IR</i>	336
<i>Vector Space Model</i>	337
<i>Probabilistic Model</i>	339
Nontextual IR	339
<i>Concept-Based Indexing: Textual Descriptors and Metadata for Multimedia</i>	341
<i>Content-Based IR for Multimedia</i>	342
Types of Searching	343
Summary	344
References.....	344
Additional Reading	345
 15—Internet Search	347
The Challenges of Internet IR	348
<i>The Size and Currency of the Web</i>	348
<i>Format Issues</i>	349
<i>Quality of Information</i>	350
<i>The Invisible Web</i>	350
<i>Web Organization</i>	351
A Short History of Internet Search.....	351
Internet Search Services.....	353
<i>Directories</i>	353
<i>Search Engines</i>	355
<i>Metasearch Engines</i>	356
The Economics of Search Engines	356

Index Size and Indexical Bias	358
Internet Search Engine Anatomy	359
<i>The URL Database</i>	359
<i>Creating the Index</i>	360
<i>The User Interface</i>	361
<i>Query Processing and Relevance Ranking</i>	361
<i>Link Analysis and Retrieval Rankings</i>	362
Google: An Example of a Large-Scale Search Engine....	363
<i>Google History and Background</i>	364
<i>Google Infrastructure</i>	365
<i>Google Architecture</i>	365
<i>Google and Relevance Ranking</i>	367
<i>PageRank</i>	367
<i>Google Revisions and SEO</i>	369
Semantic Search	370
<i>Google's Knowledge Graph</i>	373
<i>Google Snippets</i>	374
<i>Bing Smart Search</i>	375
Peer-to-Peer	375
Clustering and Visualization.....	376
Smart Agents, Personalization, and Privacy	377
Specialized Search Engines	378
Comparison with Traditional Online Services	379
Summary	381
References.....	382
16—Libraries and the Internet: Learning from the Past, Exploring the Future	387
The Internet: Pre-Web.....	390
Web 1.0.....	390
Web 2.0.....	391
Web 3.0.....	392
<i>Virtual Worlds</i>	393
<i>The World Wide Computer</i>	394
<i>Semantic Web</i>	395
<i>Not Dead Yet</i>	396
Library 2.0: Working in the Web	396
<i>The Googlization of Information Seeking</i>	397
<i>Library Systems</i>	398
<i>The Integrated Library System</i>	398
<i>The Library Catalog</i>	399
<i>Open URL and Link Resolvers</i>	402
<i>Discovery Tools</i>	402
<i>Tagging</i>	403
<i>The Library as Place</i>	404
<i>Social Media</i>	404
<i>Social Networking Sites (SNS)</i>	406
<i>Instant Messaging (IM) and Twitter</i>	408
<i>RSS Blogs and Podcasts</i>	409

xvi Contents

<i>Wikis</i>	409
<i>Sharing Photos, Videos, and Bookmarks</i>	410
<i>Mashups and APIs</i>	411
Mobile Technologies and Cloud Services	412
<i>Mobile Technologies</i>	412
SMS	413
QR Codes	413
Cloud Services	415
Critiques of Web 2.0 and Library 2.0	415
Summary	418
References	418
Additional Reading	423
Appendix 1: The Binary Machine	425
Appendix 2: Web Hosting	431
Glossary	435
Bibliography	455
Index	479

Preface

In the preface to the first edition, I posed the question “Why another book about the Internet?” The answer I gave then still applies today: although there are many excellent books on networking, the Internet, HTML, Web design, Web programming, XML, and Web searching, there remains a need for a single survey text that explores these topics holistically in the context of the knowledge and skills needed by those preparing to enter information technology (IT) intensive fields such as library and information science (LIS), business and management information systems (MIS), and decision science (DIS). Many professions are increasingly dependent on these technologies, and this second edition is intended to serve as a text for a survey course introducing these interrelated Internet technologies and applications. It provides an overview of how they work, how they are evolving, and why they are important. My goal is to support courses in LIS or other disciplines that are aimed at students who are not technology experts but who find their chosen field mandates an understanding of many technical topics that they may not have encountered in their previous studies. I have drawn on 20 years of experience teaching technology topics to nontechnical audiences at a level that sufficiently addresses the subject without overwhelming those lacking a background in IT. Thus, this book provides an overview of these technologies and builds the foundation necessary to enable the student or practitioner to explore them more fully in subsequent courses or in professional practice.

Obviously, because all these technologies have themselves been the subject of entire books, it is clear that this text cannot reflect the full depth to which each could be explored. There are places where this text is deliberately more “horizontal” than “vertical” and therefore tends to be a “mile wide and inches deep” (well, perhaps “yards” deep). Such an approach is in keeping with both its survey nature and its intended nonexpert audience. A major challenge for

such a broad text is deciding what should be included or excluded. One has to decide on a reasonable starting point and boundaries. I should emphasize that this is not necessarily intended to be a text about all important library technologies, but is intended instead to focus on those I defined as primarily Internet-related technologies. The boundaries between these notions are increasingly blurred; there are many important technologies that can be accessed or enhanced via the Internet that I have chosen not to include in this text. In addition, discussing Internet technologies requires a solid understanding of basic computer and operating system concepts, which are assumed, so the reader without that background may need to seek other sources on those fundamental concepts. Finally, I should acknowledge that most of the examples provided are drawn from the Microsoft Windows environment. This is not meant to imply anything other than my personal preference. Both the Mac OS and Linux could have been used instead—the choice is simply due to the fact that most of my personal computing experience has been in a Windows-centric environment.

In my field of Library and Information Science, there is a clear need for a broad, holistic understanding of these Internet technologies. More than a decade ago Roy Tennant described a “Digital Librarian Shortage,” observing that although not everyone needs to be able to code software, “they should know what software is capable of doing, when a program could be easily written to accomplish a task, and what skills someone needs to write one” concluding such knowledge was crucial to professional success (2002, p. 32). Although his opinion regarding the need for all librarians to be conversant with HTML has since changed, he acknowledges those with these skills will have additional career opportunities (Tennant, 2013). Many in the profession still find “learning to speak machine” useful for many activities such as tweaking the PHP code in a WordPress template, performing a batch edit on a group of MARC records, or working with a catalog API (The year the code broke, 2013). There has been a surge of interest in classes devoted to learning the language of the Internet to support Web and app development predicated on the belief that these skills are both necessary and valuable in an increasingly Internet-dependent world (Wortham, 2012). I agree, believing that even as these technologies become easier to implement and apply, there will remain a need for those with a deeper understanding of what is happening behind the scenes to troubleshoot problems and apply new technologies. These skills remain relevant today, not just to librarians but also to all preparing for careers dependent on the extensive use of IT. This text addresses these topics from that point of view, acknowledging that as technologies become easier to implement not everyone needs to become expert with every technology, but that understanding what they are, how they work, and what they are capable of doing is still critical to future professional success for many.

This second edition has been somewhat reorganized and some chapters of the first edition have been combined or partially deleted to accommodate new material. Major additions to this edition include more discussion on the mobile Web, HTML5, responsive design, and content management systems. It remains organized around three main areas: Part 1 covers Internet history and administration, computing and information technology fundamentals and trends, networking and connection technologies, Internet protocols, and cyber

security. Part 2 covers building the Web and includes chapters on Web design, Web publishing and graphics, HTML, CSS, Web script programming, XML, and content management systems. Part 3 covers Internet content formats, an overview of information retrieval (IR) models and issues, and Internet search engines. Part 3 concludes with a discussion of the future of the Web and the ongoing impact of Web 2.0 and mobile technologies on the information professions. For those who might benefit from additional background on basic computing concepts there are appendices with a brief primer on binary numbers and the binary nature of digital computers and UNIX hosting issues. A glossary of terms is also provided.

The prediction I made in the preface of the first edition was that I was much like the potential buyer of any technology who knows that they can always find something better, faster, and cheaper if they wait until tomorrow. I enter this effort again knowing that like that buyer, tomorrow there will be something new I would have wanted to include. However, I believe the foundations and trends discussed in this edition will remain relevant even as new technologies continue to emerge. One of the goals of this text is to develop a frame of reference and the “eye for detail” that these future technologies will demand. The desired outcome of this process is an enhanced understanding and appreciation of these technologies that will provide the knowledge and confidence needed to support the lifelong learning expected of all working in these rapidly changing fields.

The conventions used in the text include the use of italics (except for their occasional use for simple emphasis) to denote terms found in the glossary and the use of bold font to represent commands, filenames, and tag references within the narrative. In addition to the sources I consulted for this text, each chapter may list other recommended sources or websites for those desiring more information on the topics discussed.

REFERENCES

- PLA Contributor. (2012, October 25). 2012: The year the code broke. The Wired Library. Retrieved May 17, 2013, from <http://publiclibrariesonline.org/2012/10/2012-the-year-code-broke/>
- Tennant, Roy. (2013, March 5). Why you should not learn HTML. The digital shift. Retrieved March 15, 2013, from <http://www.thedigitalshift.com/2013/03/roy-tenant-digital-libraries/why-you-should-not-learn-html/>
- Tennant, R. (2002). The digital librarian shortage. *Library Journal*, 127(5), 32.
- Wortham, Jenna. (2012, March 27). A surge in learning the language of the Internet. Retrieved May 6, 2013, from http://www.nytimes.com/2012/03/28/technology/for-an-edge-on-the-internet-computer-code-gains-a-following.html?_r=0.

This page intentionally left blank

Acknowledgments

A colleague once compared writing a book to giving birth. I am somewhat skeptical of this comparison, but continuing with that metaphor, it does appear to “take a village” (at least in my case). I am extremely grateful to those who encouraged, assisted, and otherwise supported my efforts in writing both editions of this book. Several colleagues provided helpful reviews that shaped and sharpened my thinking about topics that are in their areas of expertise. Professors Kwan Yi and Sujin Kim offered advice for the first edition on the sections regarding IR, scripting, and XML, and their assistance deserves mention. Rob Aken of the University of Kentucky libraries provided a review of the HTML chapter, and Eric Weig, librarian in Digital Library Services at the University of Kentucky provided helpful input and suggestions regarding digital repositories and OASIS. I also wish to acknowledge my editors Sue Easun for the first edition and Barbara Ittner for the second, Senior Production Editor Emma Bailey, Production Manager Lydia Shinoj, and all those at Libraries Unlimited and ABC-CLIO who guided me through this project and brought it to fruition. I am indebted to these friends and colleagues for all they did to make this a better book; for all their advice and assistance, I offer my sincere appreciation while also absolving them of responsibility for any remaining errors.

Obviously, most of this book does not reflect original research, and I have, as they say, stood on the shoulders of others who are the inventors, programmers, engineers, designers, and experts whose efforts I have tried to summarize into an instructional narrative. I cannot thank all these individuals personally, but they appear in the many sources referenced in this book. On a personal note, I would like to thank those who have influenced and shaped my career by sharing their time, knowledge, and advice. They are too numerous to list here, but there are several I would like to acknowledge who were instrumental to my career in LIS and provided opportunities that directed its subsequent

xxii Acknowledgments

path. Professor Stan Hannah, formerly of the University of Kentucky SLIS, and Professor Emeritus Thomas Waldhart both encouraged and mentored me, and I am deeply grateful for their support and friendship. Finally, I want to express my heartfelt thanks to my wife, Susan, not only for her help with this project but for inspiring me to pursue a second career in library science and her ongoing support, without which I would not be where I am today.

*Joseph B. Miller, MSLS Associate Professor, Emeritus
School of Library and Information Science
University of Kentucky*



Part

1

Internet Technologies

Part 1 is focused on Internet technologies, including chapters on general information technology issues, Internet and Web history and development, networks, connection technologies, TCP/IP and higher-level Internet protocols, and security issues.

This page intentionally left blank

Introduction

THE INTERNET AND THE CHANGING IT WORLD

Nearly twenty years ago, Microsoft's Bill Gates asserted that the popularity of the Internet was the most important single development in the world of computing since the introduction of the PC (Gates, 1995). His claim still holds today—the Internet is widely perceived as one of the most transformative technological innovations of the twentieth century, not only remaking our information world, but according to a recent survey, also shaping who we are with regard to our shared values and beliefs (Rosen, 2012). Although the effects of the Internet on culture, society, and individuals are undeniable, the focus of this text is to understand the technologies that support it. It is devoted to developing a practical and technical understanding of what the Internet is, how it works, how it is managed, and where it appears to be going since the first ARPANET packets were sent and received in 1969. Packets, the way datagrams are formed on computer networks, are discussed in later chapters. Such a broad agenda is quite ambitious and no single book can cover all these topics in depth. A reasonable starting point is to formally define the Internet, examine its history as well as the people who made critical contributions, and describe a few examples of the far-reaching impact it has had—and continues to have—on society and the broader information environment. The Internet has influenced and driven major developments and trends in the broad areas of computer technology, bandwidth creation, software development, and commerce. Many of these topics depend on a sound understanding of the basic concepts of information technology (IT) such as the use of binary numbers for addressing schemes, digital data representation, and Boolean logic. These will come up in various discussions throughout this text and Appendix 1 provides a review of these fundamentals.

4 Internet Technologies and Information Services

The quote by Bill Gates back in 1995 has been reaffirmed many times and by many experts and writers over the last decade. His quote referred to how much the Internet has shaped the world of computing and IT as opposed to broader societal effects. Similarly, the focus of this text will be on the technologies of the Internet, their evolution and convergence, and their application. However, we cannot ignore how the Internet continues to have enormous effects on the worlds of politics, global commerce, and society at-large. In the popular book *The World is Flat: A Brief History of the Twenty-First Century*, Thomas Friedman (2005) identified 10 “flatteners” in the global economy that are changing the lives and work of billions of people, and the emergence of the Internet is central to almost all of them. The impact of the Internet on library and other information-related professions has been dramatic, changing not only daily activities, services, and the professional preparation of librarians, but the very notion of the library. Indeed, no information provision business has remained unaffected by the Internet and the related technologies that have developed within and around it. Some would assert that *the Internet has changed everything*. Thus, although the focus of this text will be the technologies of the Internet, it will also consider some of the specific ways they have created new challenges and opportunities for professional practice in libraries. That said this text is not intended to be a broad examination of all important library technologies. Those that are examined have been included because of their direct intersection with the Internet technologies being discussed. Thus, many important and useful technologies in libraries are either not mentioned or explored in any detail. For example, software for reference management such as Endnote or productivity applications, such as Excel, is widespread and important. They are both functionally enhanced when the user is connected to the Internet; citations can be searched and downloaded into Endnote or templates downloaded into Office, but they are not Internet technologies per se. Much like the game of connecting actor Kevin Bacon to other Hollywood people or events within six steps, almost all IT today can be viewed as Internet-related due to the ways it can be used, updated, or shared using the Internet. The decision to discuss some, but not all of these technologies was an editorial one and judging any of these to be outside the scope of this book is not meant to diminish their importance in libraries.

How dramatic these changes seem depends in large measure on whether you are of the “BW” (Before Web) or “AW” (After Web) generation. Those who came of age in the Web-less world are much more amazed by the changes it has wrought; those who grew up as digital natives “post-Web” take it for granted as much as electricity or running water. This is to be expected—current generations are typically unimpressed with the technological marvels that preceded them. The fact that by the late twentieth century most people in the United States could pick up a telephone and have voice communication with almost anyone in the world certainly seemed much more magical to those who grew up in the earlier era of telegrams compared to those who grew up in today’s world of ubiquitous mobile phones and Internet access. But even though we live in an ever flattening world where universal Internet access, wireless networks, mobile phones, and devices capable of storing thousands of books, songs, and photographs are taken for granted, it is still informative to step back and examine how the Internet has changed so much for so many so quickly.

THE INTERNET DEFINED

The term *Internet* occurs throughout this text, so it is important to begin by defining it. In a generic sense, the term *internet* refers to any group of interconnected networks. However, in this text, *Internet* refers to the global network of computer networks utilizing *packet switching* and the Transmission Control Protocol/Internet Protocol; TCP/IP) standard, which includes the family of supporting services and protocols examined in detail in other chapters.

A BRIEF HISTORY OF THE INTERNET

The modern Internet began with the formation of the ARPANET in 1969 by the Advanced Research Project Agency within the Department of Defense. Although earlier network and computer technologies were essential to development of the Internet, most people identify the ARPANET as the seminal event in its history. The ARPANET initiative resulted in the first packet switching computer network connecting a computer at UCLA with one at Stanford University, thereby creating a two-node network. It was from this modest beginning that the modern Internet emerged. ARPANET in part was a response to national defense concerns and a desire to build a decentralized communication network potentially capable of surviving the loss of a major hub and the electromagnetic disruption of a nuclear strike. ARPANET interconnections utilized dedicated processors called Interface Message Processors (Leiner et al., 1997).

While the origins of the Internet go back to the 1969 ARPANET test bed, because TCP/IP is so integral to a definition of the Internet, one could argue that the Internet of today really began with the adoption of TCP/IP by ARPANET on January 1, 1983. This foundation protocol suite is the subject of Chapter 4. Zakon (2011) provides a complete timeline using Hobbes' Internet Timeline; an abbreviated list of selected significant events since TCP/IP was defined is given here.

- 1980** TCP/IP standards are defined. The modern Internet (as well as numerous private intranets) is built on this family of protocols and services.
- 1983** TCP/IP protocols are implemented by ARPANET. The MILNET is separated from ARPANET. The Internet Activities Board (IAB) is established.
- 1986** The National Science Foundation (NSF) connects the five new supercomputer centers into a network known as the NSFNET. These supercomputers were very expensive and in high demand, and the NSFNET was created to facilitate access to these centers across the country. This network used the same foundations and infrastructure of the then ARPANET. The NSF had strict guidelines for what constituted appropriate uses of this network. It prohibited commercial activities, which gave rise to a culture that still influences the Internet community even though it is clearly now highly commercialized. For instance, the open source software

6 Internet Technologies and Information Services

movement has some of its roots in the noncommercial culture of the Internet.

- 1989** Tim Berners Lee proposes a new information management system at CERN (an acronym derived from Conseil Européen pour la Recherche Nucléaire). His superior, Mike Sendall, describes it as “vague, but exciting” and approves further development (CERN, 2008). His proposal would evolve to become the World Wide Web.
- 1990** The overlapping infrastructure and uses of the NSFNET and ARPANET result in the ARPANET disappearing as a separate entity, being supplanted by the NSFNET.
- 1991** The High Performance Computing Act is approved by Congress. This legislation, introduced by Senator Al Gore of Tennessee, authorizes the National Research and Education Network (NREN). Al Gore’s involvement with this legislation gave rise to the “Al Gore invented the Internet” mythology. Gore actually, never made that claim, but he was an early and prominent national voice regarding the potential of the Internet for research and education. Although NREN was legislatively approved, it did not become a separate networking entity; it did influence thinking and policy decisions about the role of government in supporting Internet access for educational and research purposes.
- 1991** The World Wide Web (WWW, aka the Web), developed by Tim Berners-Lee at CERN goes live. The Web really provided the first glimpse of the Internet to many. Berners-Lee, a consulting programmer, initially focused on the problem of document sharing within a specialized community, namely, the physicists at CERN. As he explored a hypertext solution to this problem, he recognized it could have far broader implications for information sharing via the Internet. With the support of CERN, he went on to develop many of the protocols of the Web as well as the markup language needed to create Web content. The rest, as they say, is history (Berners-Lee, 1999).
- 1993** The MOSAIC browser is created and released by a team of programmers working at the National Center for Supercomputing Applications (NCSA) at the University of Illinois, Urbana-Champaign. The team included Marc Andreessen who would go on to form and lead Netscape Corporation. The development of this user-friendly, graphical browser brings the Internet and the emerging Web to a mass audience.
- 1995** The original NSFNET has evolved into an ever more commercialized Internet, so NSFNET reverts to its research roots and becomes a very high-speed network, again devoted to education and research. It will give rise to the Internet2 initiative.
- 1997** Internet2 is developed as a nonprofit consortium of over 170 U.S. universities and institutions. Its purpose is to support research and be a test bed for new networking technologies.

- 1998** Internet Corporation for Assigned Names and Numbers (ICANN) is formed as a nonprofit private corporation with responsibility to coordinate the stable operation of the Internet in four key areas: the Domain Name System (DNS); the allocation of IP address space; the management of the root server system; and the coordination of protocol number assignment. For much of its existence, ICANN is a nonprofit corporation that reported to the U.S. Department of Commerce under a Joint Project Agreement, but there has been international pressure to reformulate this organization under an international umbrella such as the United Nations. The U.S. Department of Commerce oversight of ICANN will end in 2015 and be replaced by a multinational group of stakeholders to be proposed by ICANN (Farrell, 2014).
- 1998** Google incorporates and launches its search engine.
- 2000** Internet2 backbone network deploys IPv6. ICANN releases new Top-Level Domains (TLDs) including .aero, .biz, .coop, .info, .museum, .name, and .pro. Web size estimated at 1 billion findable pages.
- 2003** Originally used in a book title by Dermot A. McCormack in the context of ecommerce, the term “Web 2.0” is used by Tim O'Reilly and John Battelle to describe the interactive, next generation Web. Blogs and social media are becoming popular.
- 2004** ICANN authorizes generic Top-Level Domains (gTLDs) of .asia, .cat, .jobs, .mobi, .tel, and .travel. Web 2.0 gains traction with social networking and Web mashups.
- 2008** IPv6 addresses added to six root zone servers. Google's crawler reaches 1 trillion pages.
- 2010** Apple introduces the first iPad and smartphone shipments overtake those of the PC. *Wired* magazine cover declares, “The Web Is Dead.”
- 2013** Worldwide devices (PCs, tablets, and mobile phones) could total 2.4 billion units, up 9 percent from 2012; PC shipments declined 7.6 percent but is offset by strong gains in the mobile market (Gartner says, 2013).

ADMINISTRATION OF THE INTERNET

The Internet is a complex information system heavily dependent on strict adherence to standards. It is global in reach, crossing all political and geographical boundaries, yet amazingly, there is no centralized governing authority in charge of its overall administration. It is somehow fitting that an entity that has resulted in a flattened, interconnected, and collaborative world is itself managed in a similarly decentralized and cooperative fashion. Internet administration involves interaction among international standards organizations, the private sector, and government agencies. The Internet Society (<http://www.isoc.org>) is an umbrella organization of hundreds of groups and thousands of individuals

8 Internet Technologies and Information Services

engaged in developing technologies and standards to maintain the viability and global scalability of the Internet. Examples of groups within the umbrella include standards groups such as the Internet Engineering Task Force (IETF) and Internet Architecture Board (IAB). A progression of different entities has been responsible for the critical area of domain space control. The first was the Network Information Center (NIC), the official provider of Internet domain name registration services to the world for registration of domain names in the .COM, .NET, and .ORG top-level domains (this is now an ICANN affiliate). InterNic (<http://www.internic.net>) provided the public with information regarding Internet domain name registration under the auspices of the U.S. Department of Commerce. The Internet Assigned Numbers Authority (IANA) provided central control for domains, addresses, and protocols. Currently, ICANN (<http://www.icann.org>), a nonprofit corporation, has taken over the role of naming and address space management and IANA is a department within ICANN. Another important body is the World Wide Web Consortium (W3C), which is responsible for developing and maintaining Web standards such as Hypertext Markup Language (HTML) and Extensible Markup Language (XML).

The first Internet backbone service was run on leased telephone lines and it still depends on telephone and cable networks. These include services controlled by UUNET (now part of Verizon), AT&T, and Sprint; they provide the Internet Exchange Points (IXPs) that make the physical connections among the interconnected networks. These companies, along with the cable Internet service providers (ISPs), are responsible for the “last mile” connection to homes and businesses. They also work closely with the various governing bodies to set policy. It is not hard to imagine that the various players involved in Internet policy and oversight have differing agendas that sometimes require government intervention to resolve. One example of the intersection of government regulation, policy groups, and market forces is the issue of *net neutrality*. This is primarily a policy issue with potentially enormous implications for different users and the cost of their Internet activities, so it is not surprising that the involved parties hold differing views.

Net Neutrality

The core idea of net neutrality is that all packets are equal; no packets are given “priority” status on the Internet over others. The controversy is whether this should remain the case or if there should be packet-level discrimination for some services. There are those who believe that in order to maintain quality service for new applications such as Voice over Internet Protocol (VoIP), video streaming, or interactive games, high-speed Internet “lanes” should be available at the expense of net neutrality. Much of the pressure for this position comes from the telecommunications and cable companies. On the other hand, many search engines and software companies favor the current status quo of de facto net neutrality. The argument is also philosophical; some maintain mandating “net neutrality” would increase undesired government intervention with Internet bandwidth, while others believe abandoning net neutrality could result in less technical innovation. There is some precedent for supporting the premise of net neutrality within the umbrella of telecommunication law and previous

government common carrier regulation. In 2010, a Federal Communications Commission (FCC) ruling created two classes of access, one for fixed line providers and the other for wireless, which resulted in what some call net “semi-neutrality.” They also established rules that wireline providers could not block or discriminate in transmitting lawful content and services to their customers. However, the courts have not sided with the FCC to date, initially ruling in 2010 that the FCC lacks authority to mandate net neutrality (Wyatt, 2010). A Washington, D.C. Appeals Court ruling in 2014 upheld the FCC’s authority in this area but struck down much of the 2010 FCC Open Internet Order antiblocking and antidiscrimination rules, primarily because the commission had previously chosen not to treat broadband providers as common carriers. However, the ruling left the disclosure rules intact, so an ISP that gives certain data preferential treatment must disclose that fact to its customers (Chacos, 2014). In a recent development, the FCC has released draft new rules that will allow content providers to pay for faster connections, which means Comcast or Verizon could charge companies like Netflix or Google different rates for priority service (Wyatt, 2014). It appears the telecoms and cable operators are winning this round of the debate.

There are precedents for a “pay as you go” model—early dialup connections described in Chapter 3 often were associated with per minute charges for connect time, and mobile data services often charge for data downloads that exceed some monthly amount. Changes in how the Internet is being used have also influenced this debate; for example, a 2011 report estimated Netflix accounted for 32 percent of all downstream Internet traffic (Wasserman, 2011). Fairness is often used to support both sides of this debate—for example, some believe users should pay for bandwidth-intensive uses. On the other side, others believe there is an inherent conflict of interest when Internet providers sell their own competing content. It will be interesting to watch how this policy debate develops over time.

IT FOUNDATIONS

In this text, a basic understanding of the computing technologies the Internet depends on is assumed; however, depending on the depth of that IT knowledge, it may be necessary to review these topics in other sources. The following is an overview of the topics that should be part of the working knowledge base needed to understand many of the technologies described throughout this text.

Computers: The Binary Machine

Computer technology is the core engine of all IT. The modern computer is an electronic, digital device that can transform inputs into outputs under the control of a stored program. Throughout this book, the terms *computer* and *personal computer (PC)* are used somewhat interchangeably. Obviously, many types of computers exist, and the PC is typically a desktop or laptop computer. In addition, while the term *PC* is most often assumed to refer to Wintel systems, it can refer to any type of personal computer (Windows, Linux, or Apple). The term *PC* is used generically in this book, but unless otherwise noted, all

10 Internet Technologies and Information Services

the specific examples provided in various figures or examples are based on the Wintel platform (i.e., Intel processor architecture running a version of Microsoft Windows). This Windows-centric perspective simply reflects that most of my computing work has been in a Windows environment, but there are certainly other important platforms based on the Mac OS or Linux that could have been used in these examples. The notion of a personal computer has been extended with the emergence of the many tablets and smartphones, which are increasingly becoming the preferred platform for many.

One could argue that the path to the tablet computers of today began as early as the 1830s when Charles Babbage was designing and attempting to build his mechanical computer. Although brilliant, Babbage lived in the wrong century—mechanical devices could never achieve the speed and power of electronic ones. The first electronic computers were analog ones. Vannevar Bush of Memex fame worked on electronic analog computers in the 1920s¹ and in the late 1930s, the Atanasoff-Berry Computer led to the World War II-era Electronic Numerical Integrator and Computer (ENIAC). However, the modern computer we know today is a digital machine. The processors that are the “brains” of the computer on a very basic level are just extremely fast switching devices with many millions of transistors. Although there is still interest in analog computers or digital-analog hybrids, digital technologies are far more prevalent, making it worthwhile to examine the distinctions between analog and digital technologies.

Throughout the twentieth century, the “electronics revolution” was an analog one and included the modern marvels of the telephone, phonograph, radio, and television. These analog technologies depend on electromagnetic waves as carriers of information. Waveforms have both amplitude and frequency; *amplitude* is a measure of the height of the waveform, and *frequency* is a measure of the number of cycles per second of a waveform. Wavelength is the distance from one peak to another and is inversely proportional to frequency; that is, shorter wavelengths have a greater number of cycles per second, as shown in Figure 1.1.

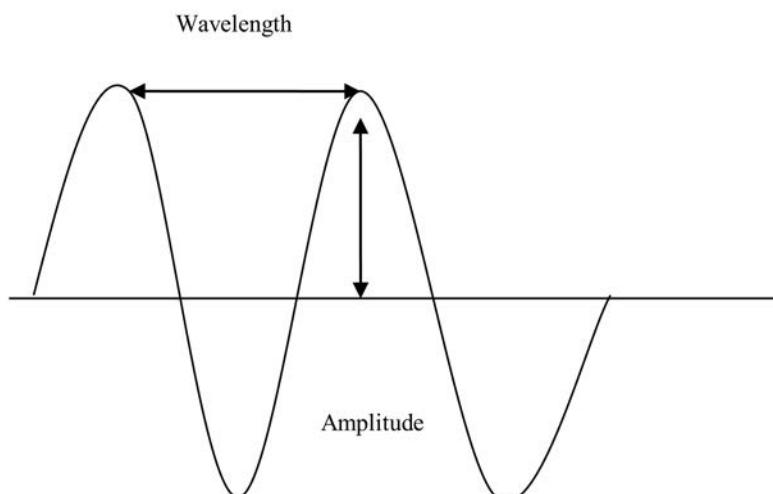


Figure 1.1 Waves have lengths and amplitude. Frequency is measured in cycles per second and is inversely proportional to wavelength.

Information can be added to the carrier signal through signal modulation. For instance, AM radio depends on *amplitude modulation* where the information is added by varying the wave amplitude, shown in Figure 1.2. FM radio depends on *frequency modulation* where the wave frequency is varied, as shown in Figure 1.3. Phase modulation refers to a delay in the natural cycle of alternating current waveforms as a way to carry signal information. Another possibility is pulse code modulation (PCM) where the waveforms are interpreted to represent two states, which is an analog way to carry digital data.

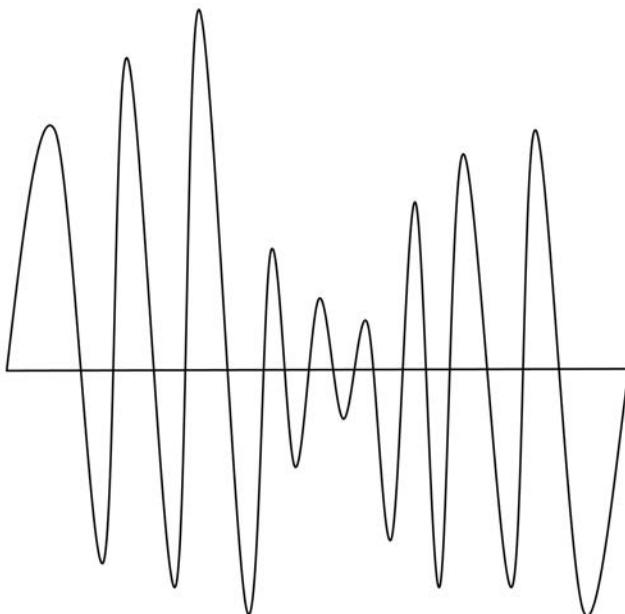


Figure 1.2 With AM, a source modulates the amplitude of a carrier wave.

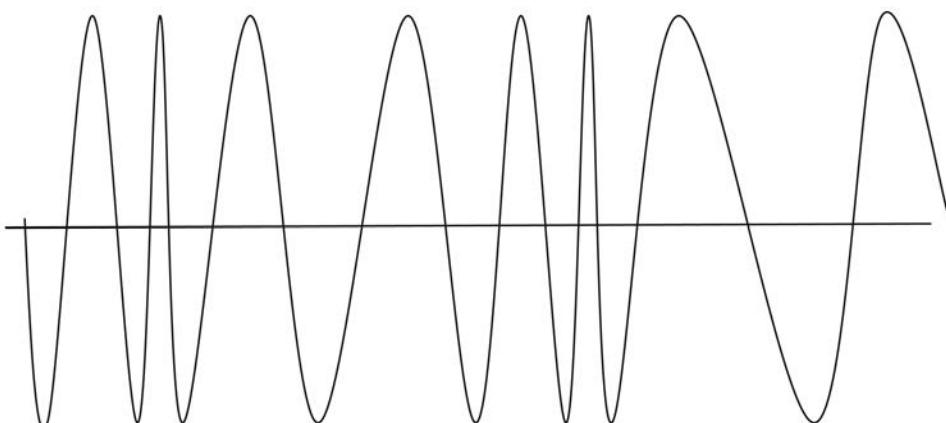


Figure 1.3 With FM, a source modulates the frequency of a carrier wave.

12 Internet Technologies and Information Services

In contrast to the continuous waveforms of analog data, digital data are represented in discrete increments. Computers and digital systems use binary numbers (0 and 1) to represent digital data. Analog to digital conversion uses sampling technologies to represent the analog signal as discrete values; the sampling level determines how faithfully the analog content is captured. In the late 1940s, Bell Laboratories mathematician Claude Shannon and others were working on the mathematics of sampling technologies. Bell Labs had a practical interest in sampling as it applies to voice communication systems. Analog to digital conversion requires a continuous time signal conversion to a discrete time one, which can then be used to reconstruct the original continuous signal when needed. The mathematics are complex, but the Nyquist-Shannon Sampling Theorem proved that a sampling rate equal to at least twice the bandwidth of the signal could adequately reconstitute the original by allowing the interpolation of data points between samplings during conversion (Shannon, 1949). The theorem name recognizes earlier related work by Harry Nyquist, also at Bell Labs. Inadequate sampling rates accentuate *aliasing*, which is a distortion or artifact of the sampling process. Sampling techniques are the basis of the many tools and programs developed to create digital representations that are very close to the original analog data. For instance, with commercial music CDs, a sampling rate of the original analog music of 44,100 times per second results in digital playback that is indistinguishable from the original to most human ears (Watkinson, 2000).

The modern computer is a binary, digital device. The computer is known as a binary machine for two reasons. First, computers use *binary numbers* to represent instructions and data. Second, computers are binary machines because the logic circuitry depends on a binary logic system known as *Boolean Logic*. Analog computers are possible but represent a more complex engineering problem than digital computers. Digital computers need only represent two possible states, which are then interpreted as the needed 0s and 1s. Representing two states is easier to engineer with simple switches or as the presence or absence of a magnetic charge on a disk. Within the computer, data and instructions in the form of bits move from one location to another using a *bus*, or circuit path. The number of bits moving in tandem depends on the width of this data bus, always in multiples of 8 bits; it could be as narrow as 1 byte (1 byte is 8 bits) and as wide as 8 bytes in earlier PC architectures. Network and packet technologies are now being employed at the hardware level of the PC with the development of the *Peripheral Component Interconnect Express* (PCIe) architecture that creates a point-to-point data bus that uses a serial stream of 8 bit packets instead of the previous parallel stream bus architectures. Digital data streams can be transmitted over analog connections with a *modem*, which stands for modulator/demodulator. This device modulates the computer's digital stream into an analog form. The demodulator in the receiving computer converts the analog signal back to a digital one. Modems and other connection technologies are discussed in more detail in Chapter 3.

Knowledge of these fundamental ideas is useful, not only as they apply to computing technology, but in a variety of other IT contexts in this text. For instance, binary numbering is relevant to topics such as how data are represented with text codes and why Internet protocols were developed for text standards. Understanding binary numbers is essential to the topics of IP addressing,

subnet masks, and the encoding schemes needed to convert binary data to a text representation. (See Appendix 1 for additional discussion of binary numbers.)

Client–Server Architecture

Another important concept is *client–server architecture*, an essential structure to almost every activity that takes place on the Internet. This important idea can be confusing depending on the context in which the terminology is used. Client–server architecture utilizes two pieces of software designed to work together: the client’s job is to formulate a request to the corresponding server software, which in turn responds to the request. This correspondence usually takes place via a specific communication channel known as a *port*. On the Internet there are many such software pairings communicating with each other: mail clients with mail servers; telnet clients with telnet servers; FTP clients with FTP servers; and Web clients with Web servers. Often each of these is on hardware dedicated to the server function, but it is the software that is critical to the definition of this architecture. A desktop PC could be running both client and server programs concurrently, thereby performing both roles.

THE INTERNET AND TECHNOLOGY TRENDS

The Internet has driven huge changes in how computing and IT are used, affecting the knowledge base required by those planning to create new information services and instruct others in their use. Over the last 25 years, there have been dramatic changes in computer and storage technologies, bandwidth availability and capacity, and the use of the Internet in business and for human collaboration. Discussion of some specific Web-related trends is deferred to subsequent chapters, but a few broad megatrends are explored here. The first of these core changes is how users now perceive the computer itself.

Computing to Connecting

The Internet is largely responsible for a major paradigm shift regarding computer technology that has occurred during the life span of the PC, a shift from “crunching” to “communicating.” When computers were first developed, they were viewed primarily as computational devices able to do large scale, complex number crunching and symbol manipulation to support business, government, and science. Early computers were used during World War II for calculating projectiles, code breaking, and the Manhattan project. Even if IBM’s Thomas Watson never really predicted a world market of “maybe five computers” at that time, that sentiment has some truth, as early computers were extremely complex and expensive.

This was still the predominant view of computers when the PC was introduced and it was primarily marketed to small businesses and early adopters. These early users needed to be quite enthusiastic about the technology, as

14 Internet Technologies and Information Services

first-generation PCs were expensive and not particularly user friendly. During the last several decades, the cost of computing has declined significantly, and power has increased dramatically. These trends are consistent with the expectations of Moore's Law,² which has been extrapolated to most everything related to computing. For instance, in 1971, the Intel 4004 had approximately 2,300 transistors and was capable of about 60,000 instructions per second. By 2005, Intel reported P4 Itanium processors with 410 million transistors performing about 10,800 million instructions per second (MIPS) or about 10.8 billion instructions per second (Friedman, 2005). By 2012, the Intel Xeon Phi had 5 billion transistors capable of one teraflop (10^{12} floating-point operations per second; Intel® Xeon Phi™, n.d.). The change has been equally dramatic when the capacity and cost of storage are considered. The first PC hard drives were 10 megabytes (MB) and added about \$1,000 to the price. Now a 500 GB drive costs about \$50, providing 50,000 times the capacity for a twentieth of the cost. Even with computing power increasing and prices declining, PC sales did not really accelerate until the emergence of the Web in the early 1990s. According to the U.S. Census, the number of homes with computers grew from 15 percent in 1990, to 56 percent by 2001, and to 76 percent by 2010. People connecting to the Internet from home in the United States went from 18 percent in 1997, to 54.3 percent in 2010 (U.S. Census, 2006, 2010). Factors such as lower prices and better operating systems certainly contributed to the rapid growth in the PC market, but much of their acceptance and use was, and still is, related to accessing the Internet. The Internet is now driving a similar shift to mobile devices and tablets.

Bandwidth

An essential requirement for the computer to become an effective communication device is the availability and development of a connection infrastructure with sufficient bandwidth to support the associated activities and resulting volume of packets. *Bandwidth* is a measure of the capacity of a medium to carry a signal. Bandwidth developments have experienced the same type of self-reinforcing cycle computer technology has; namely, greater demand has resulted in more technological innovation, higher capacity, and lower cost, which creates more demand. The Internet was made possible by an existing telephone network infrastructure available throughout the developed world. Up until 1991, the original NSFNET backbone service used leased T1 lines that could only carry 1.5 Mbps (millions of bits per second). In the "BW" Internet, the main backbone service of the entire Internet had far less bandwidth than typical home Digital Subscriber Line (DSL) or cable broadband Internet connections have today. The high-capacity bandwidth of modern broadband enables services such as VoIP, audio and video streaming, and real-time video conferencing. There continue to be huge advances in these connection technologies; for instance, fiber optic technologies are being developed to carry 1 terabit per second on a single fiber strand, which is hundreds of thousands of times the capacity of the original T1 lines. Wireless networking and mobile data services are also contributing to the

increase in bandwidth, eliminating many of the barriers associated with a wired environment.

Going Mobile in a Post-PC World

Wireless connections and powerful mobile devices are making Internet access without tethering network connections the norm resulting in the Internet being more fully integrated into daily life. The shift toward mobile devices and smartphones has been one of the most dramatic trends of the last decade. Apple introduced the first iPhone in 2007 and it was hugely successful, setting the standard for other smartphones to come. Apple followed with the introduction of the iPad in 2010 and it was an immediate success, selling 300,000 units on the first day of release and more than 15 million by the release of the iPad2 (Apple iPad launch marred, 2010; Apple launches iPad 2, 2011). Smartphone shipments overtook those of the traditional PC in 2010, and Nielsen reports that as of mid-2013, smartphones account for 61 percent of all U.S. mobile users (Mobile majority, 2013). By the end of 2013, tablet computer sales topped PC sales for the first time (Satariano, 2013). These devices and the 3G/4G services they utilize have altered the Internet landscape significantly, changing both how people access Internet content and how designers create and manage websites. The impact of mobile devices used for Internet access will be addressed in various contexts in this text.

Another feature of mobile Internet access is the way it can be combined with *geolocation*; the information about your physical, real-world location. Geolocation information can be associated with an IP or MAC address and can be shared and utilized; it can allow other people or applications to use location data to see how nearby your friends are, get directions to the restaurant you are reading about, or to geotag the photos you take.

There's an App for That

The success of mobile depends in large measure on an easily available and huge selection of inexpensive, or often free, “apps.” Generically, “app” is short for “application”—the term used for almost all computer software. However, the term app is also associated with the idea of “appliance-centered” computing that makes the device a very effective, albeit sometimes more limited, tool to accomplish a task. Thus, apps today tend to be much more task-specific; they are usually nimble solutions to a specific need designed to run on less powerful and smaller screen mobile devices. Much of the success of the Apple iPhone and tablets has been the rapidly growing collection certified apps available from the Apple App Store, now totaling more than 775,000 apps (Costello, 2013). The Android (now Google Play) and Windows markets have their own libraries to support those platforms. There are a wide variety of apps for every category of user need—apps for messaging and social media, utilities and productivity, music and entertainment, shopping, games, sports and health, and news and magazines. In 2013, the overall use of mobile apps rose by 115 percent from the previous year, led by a 203 percent increase in social media app use (Khalaf, 2014).

The Cloud

Working on the Internet has become synonymous with working in the “cloud.” Cloud computing is the intersection of virtual machines, remote services for hardware and software, and ubiquitous access to the Web as a platform, often with a mobile device. Working in the cloud can mean simply using a remote host for data storage such as Google Drive, Apple’s iCloud, or Microsoft’s OneDrive (previously called Skydrive), using Web-based applications via a browser, or running a full virtual machine on a site like Amazon. Cloud solutions offer secure and cost-effective IT services that are becoming increasingly important to all organizations, including libraries. Cloud applications are discussed further in Chapter 2.

Big Data

Internet activities such as social media use, cloud storage, website-tracking data, and ecommerce transactions along with traditional scientific output and government information all are generating huge global data sets that can be mined to identify trends or find new collective meaning. All these data are often referred to as “big data.” Gartner defines it as data that are “high-volume, high-velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making” (Big data, 2013). The amount of data in various digital repositories is measured in mind-numbing numbers; Google has huge data centers, estimated to have many petabytes (a petabyte is a million gigabytes) of information. However, that is dwarfed by the data stores of the National Security Agency, estimated in yottabytes (a yottabyte is a billion petabytes; Trenholm, 2009). Such huge data sets are difficult to collect and analyze, and companies are spending large portions of their marketing budgets on analytics to use these data strategically (Moorman, 2013). For libraries, there are low-cost options such as using Google Analytics to make Web design decisions or using one of the social media monitoring tools to track user opinions and views.

Going Open Source

The open source software movement is an interesting trend with broad implications for many organizations, including libraries. Open source is a model of software development that provides free access to programs created and supported by developers and users themselves. The Internet has played a significant role in the success of this model by fostering a culture of open access, facilitating virtual communities and collaboration to develop and support complex applications, and providing a global distribution framework for the resulting programs. The notion of open source being free does not imply there are no economic costs because there are costs in terms of people or fee-based support needed to implement open source solutions. However, the programs are free in that the source code is freely available to developers to download, modify, and adapt to meet specific needs. Open source solutions are important not just

in theory but because they are available to support many of the technologies discussed throughout this text, such as content management systems. Open source software such as Linux, Apache, PHP, MySQL, Drupal, and Joomla are all examples of important open source solutions that have supported Web development. A few additional open source applications of particular importance to libraries are highlighted in Chapter 16.

Internet-Based Collaboration

Another important trend is the growing use of powerful collaborative and networking tools such as social networks, blogs, news feeds and aggregators, podcasts, and collaborative content management. Really Simple Syndication (RSS) is the foundation protocol of many of these tools and is an example of XML technology. The idea of “syndication” is broadly familiar from other communication business models. For instance, a newspaper syndicate is a group of papers under the direction of a single company; in the context of RSS, it is an option to subscribe to an information source. The success of Internet in fostering social connections is also evident from the success of Facebook, which reaches over a billion active monthly users (Tam, 2013). Social networks, Wikis, blogs, twitter feeds, and Google Drive are all examples of how the Internet has created new avenues for collaborative work and they are explored further in Chapters 2 and 16.

The Long Tail

Networks and digital resources remove the physical constraints inherent in the economic models of the past, giving rise to new paradigms described by Anderson in “The long tail” (2004). This refers to how a graph of demand for consumer items (books, music, videos, etc.) shows a large peak for the popular and a gradually declining level of demand for the less popular. In the physical world, this curve would likely go to zero, as the item would become less and less available once its sales start to decline. In a retail environment with limited space, it is logical to weed less popular items in favor of more popular ones. However, in the digital world, everything can theoretically remain available because storage space constraints are not a major concern, and the global reach of the Internet increases the likelihood that a sustainable niche market exists for these items. The power of search enables an audience to find items in the “long tail” portion of the graph. This idea is visualized in Figure 1.4. Some of the effects of the Long Tail are explored in Chapter 2.

The Internet of “Everything”

When mobile devices, high-capacity wireless bandwidth, and Geographic Information Systems (GIS) are combined with exciting new apps and cheap, small sensors, we get the “Internet of everything.” This name comes from a center at Wichita State University that is based on the premise that low cost,

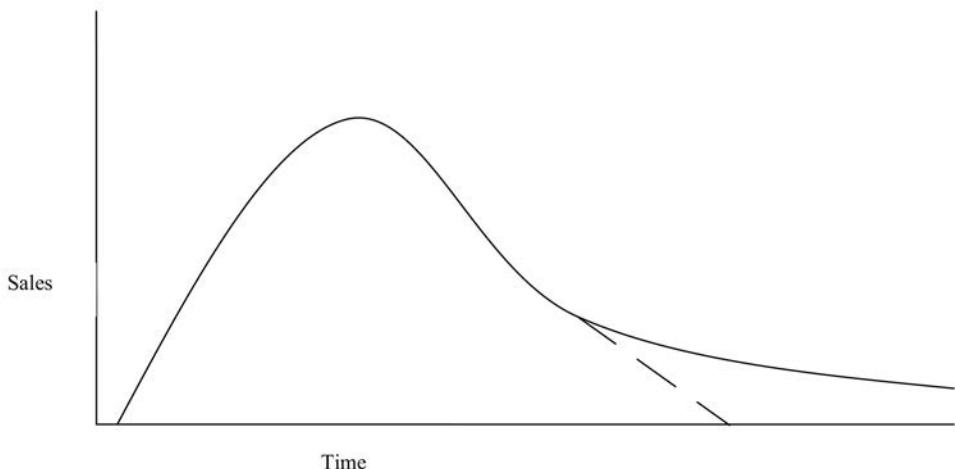


Figure 1.4 The Long Tail (after Anderson, 2004). The sales cycle of a successful consumer item will peak, but as its popularity wanes, valuable shelf space will be given to a new more popular item, and both sales and availability will decrease to zero (dashed line). However, in the digital world where the item can still be available, demand will taper off into the “long tail” portion of the graph that continues along the x-axis. This small but ongoing demand could cumulatively result in more total sales over time than that of the initial peak.

ubiquitous connectivity can be combined with apps and remote sensors to drive innovation (Wenzl, 2013). Professor Ravi Pendse points out that there are actually multiple “Internets” today. There is an “Internet of information,” that has become indispensable to information seeking and research, exemplified by tools such as Google and Wikipedia. There is an “Internet of people” with tools such as Facebook, LinkedIn, and Twitter to expand human connections and communication. The “Internet of places” combines satellite positioning, GIS, and maps that use tools such as Google Earth and Foursquare (<https://foursquare.com/>) to allow us to find physical places and each other. However, he envisions a growing “Internet of things” where billions of sensors attached to most everyone and everything will communicate with each other and us through specialized apps (Wenzl, 2013). Think about a dramatic advance in the technologies of the “OnStar” driver protection system that could use a sensor in a bicycle helmet, or even in the sidewalk itself, to call for help automatically if you are in an accident. Sensors worn by you or in your clothes could communicate health information to doctors or sensors in parking lots that could report to you about available spots as you drive to work or school. This is not just in the future, it is happening now with devices such as Google Glass that can provide real-time data about your environment or Fitbit wristbands that can monitor physical activities and sleep and then synchronize the data to your mobile device. Such monitoring goes beyond just being a novelty but would be of great benefit for many with life threatening conditions; for example, potentially monitoring blood sugar levels of diabetics or providing alerts to those with heart conditions. Google is currently experimenting with a contact lens with a sensor chip the size of

a glitter flake that can monitor glucose levels in tears (Otis & Parvis, 2014). Of course, some people fear this level of connectivity will take on the feel of a surveillance state and worry the data could be misused, for example, by an employer or insurance company. There will be concerns about privacy and security in such a world, but the “Internet of everything” seems to be well on its way.

SUMMARY

The Internet and related Web technologies have changed the world, providing both new opportunities and new challenges. It seems inevitable the pace of change will continue or accelerate, and the Internet will continue to drive much of this change. Although it does seem at times that the Internet has *changed everything*, a few qualifying points should be made about this sweeping generalization.

First, the pre-Web Internet certainly was a change agent with the *potential* to change everything, but the post-Web Internet has driven much of the actual change that matters to most people. Internet advocates in pre-Web environment sometimes used the “Field of Dreams” reference of “If you build it, they will come” to describe how they anticipated the Internet would change the world. However, much of the anticipated change did not really occur until the emergence of the Web, so it seems a more realistic statement is, “If you build it, *make it really easy to use and important to people’s lives*, they will come.” The Web is responsible for much of our perception of the importance of the Internet and in fact, few now make a distinction between them. However, they are not synonymous, and the Web is explored further in the next chapter. That said, the Internet gave rise to the Web, and Internet technologies continue to be at the center of the digital convergence emerging from the mix of cheap and powerful computers, mobile devices with geopositioning capabilities, nearly limitless bandwidth, and useful net applications. This convergence is giving rise to new generations of the Web and new opportunities for services that depend on it. No matter how one views the Internet of today, it is clear that it will continue to be a powerful engine of change.

Second, while emphasizing the positive impact of the opportunities the Internet has created for society and for libraries in particular, there has been, and will continue to be, a dark side as well. Pornography and filtering requirements have created many administrative and ethical issues for organizations providing public Internet access. Privacy and security concerns occupy much of the daily attention of Internet users, malware poses real threats, and spam, once estimated to comprise as much as 80 percent of all email, continues to be a nuisance (Zeller, 2005). Misconceptions about the value and usefulness of Internet resources; inappropriate materials; the long-term preservation of net-available digital resources; the potential for plagiarism; the cavalier attitude of some regarding copyright; and strategies for digital rights management that limit fair use are just a few of the issues that continue to challenge Internet users.

NOTES

1. In his July 1, 1945 article “As we may think” in the *Atlantic Monthly*, Bush described an electronic desktop capable of viewing a self-contained microfilm collection, bringing the universe of knowledge to the desktop, a scenario that has largely come to fruition with a PC connected to the Internet.
2. Moore’s Law is derived from the observation in the 1960s by Intel’s Gordon Moore that the number of transistors on an integrated circuit doubled every 18 months, while prices declined at a similar rate.

REFERENCES

- Anderson, C. (2004, October). The long tail. *Wired*, 12, 170–177.
- Apple iPad launch marred by technical problems. (2010). Retrieved April 11, 2013, from <http://www.telegraph.co.uk/technology/apple/7558319/Apple-iPad-launch-marred-by-technical-problems.html>.
- Apple launches iPad 2. (2011, March 2). Press release. Retrieved May 21, 2011, from <http://www.apple.com/pr/library/2011/03/02Apple-Launches-iPad-2.html>.
- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.
- Big data. (2013). Retrieved September 10, 2013, from <http://www.gartner.com/it-glossary/big-data/>
- CERN. (2008). Tim Berners-Lee's proposal. Retrieved May 8, 2014, from <http://info.cern.ch/Proposal.html>.
- Chacos, B. (2014, January 14). Appeals court strikes down FCC’s net neutrality rules. Retrieved January 14, 2014, from <http://www.pcworld.com/article/2087441/appeals-court-strikes-down-fccs-net-neutrality-rules.html>.
- Costello, Sam. (2013). How many apps are in the iPhone app store. Retrieved May 20, 2013, from <http://ipod.about.com/od/iphonesoftwareterms/qt/apps-in-app-store.htm>.
- Farrell, N. (2014, April 3). ICANN boss defends US giving up Internet ownership. Retrieved May 8, 2014, from <http://www.fudzilla.com/home/item/34393-icann-boss-defends-us-giving-up-internet-ownership>.
- Friedman, T. L. (2005). *The world is flat: A brief history of the twenty-first century*. New York: Farrar, Straus, and Giroux.
- Gartner says worldwide PC, tablet and mobile phone combined shipments to reach 2.4 billion units in 2013. (2013, April 4). Retrieved July 7, 2013, from <http://www.gartner.com/newsroom/id/2408515>.
- Gates, Bill. (1995). *The road ahead*. New York: Viking.
- Intel® Xeon Phi™ coprocessor 5110P: highly parallel processing to power your breakthrough innovations. (n.d.) Retrieved April 11, 2013 from <http://www.intel.com/content/www/us/en/processors/xeon/xeon-phi-detail.html>.
- Khalaf, S. (2014, January 13). Mobile use grows 115% in 2013, propelled by messaging apps. Retrieved January 15, 2014, from <http://blog.flurry.com/>
- Leiner, B. M., Cerf, V. G., Clark, D. D., Kahn, R. E., Kleinrock, L., Lynch, D. C., et al. (1997). The past and future history of the Internet. *Communications of the ACM*, 40(2), 102–109.
- Mobile majority: U.S. smartphone ownership tops 60%. (2013, June 6). Retrieved September 13, 2013, from <http://www.nielsen.com/us/en/newswire/2013/mobile-majority—u-s—smartphone-ownership-tops-60-.html>.

- Moorman, Christine. (2013, September 3). Big data's big puzzle. Retrieved September 10, 2013, from <http://www.forbes.com/sites/christinemoorman/2013/09/03/big-datas-big-puzzle/>
- Otis, B., & Parviz, B. (2014, January 16). Introducing our smart contact lens project. Retrieved January 23, 2014, from <http://googleblog.blogspot.com/2014/01/introducing-our-smart-contact-lens.html>.
- Rosen, Rebecca J. (2012, June 27). 59% of young people say the Internet is shaping who they are. Retrieved May 1, 2013, from <http://www.theatlantic.com/technology/archive/2012/06/59-of-young-people-say-the-internet-is-shaping-who-they-are/259022/>
- Satariano, Adam. (2013, September 11). Tablet shipments to exceed personal computers. Retrieved September 12, 2013, from http://www.bloomberg.com/news/2013-09-11/tablet-shipments-to-exceed-personal-computers.html?source=email_rt_mc_body&app=n.
- Shannon, C. (1949). Communication in the presence of noise. *Proceedings of the Institute of Radio Engineers*, 37(1), 10–21.
- Tam, Donna. (2013, January 30). Facebook by the numbers: 1.06 billion monthly active users. Retrieved May 20, 2013, from http://news.cnet.com/8301-1023_3-57566550-93/facebook-by-the-numbers-1.06-billion-monthly-active-users/
- Trenholm, Rich (2009, November 2). NSA to store yottabytes in Utah data centre. Retrieved September 24, 2013, from <http://crave.cnet.co.uk/gadgets/nsa-to-store-yottabytes-in-utah-data-centre-49304118/>
- U.S. Census. (2006). Computer and Internet use in the United States: 2003. Retrieved October 1, 2007, from <http://www.census.gov/population/pop-profile/dynamic/Computers.pdf>.
- U.S. Census. (2010). Computer and Internet use in the United States: 2010. Retrieved April 11, 2013, from <http://www.census.gov/hhes/computer/publications/2010.html>.
- Wasserman, Todd. (2011, October 28). Netflix takes up 32.7% of Internet bandwidth. Retrieved September 8, 2013, from <http://www.cnn.com/2011/10/27/tech/web/netflix-internet-bandwidth-mashable>.
- Watkinson, John. (2000). *The art of digital audio* (3rd ed.). Oxford: Focal Press.
- Wenzl, Roy. (2013, January 19). Billions of sensors power Wichita State professor's vision of interconnected world. Retrieved January 25, 2013, from <http://www.kansas.com/2013/01/19/2643201/billions-of-sensors-power-wichita.html#storylink=misearch>.
- Wyatt, Edward. (2010, April 6). U.S. Court curbs F.C.C. authority on Web traffic. Retrieved April 13, 2013, from http://www.nytimes.com/2010/04/07/technology/07net.html?pagewanted=all&_r=0.
- Wyatt, Edward. (2014, April 23). FCC will allow Internet fast lane, sidestepping 'Net Neutrality'. Retrieved April 24, 2014, from <https://www.yahoo.com/tech/fcc-will-allow-internet-fast-lane-sidestepping-net-83658252368.html>.
- Zakon, R. (2011). Hobbes' Internet timeline. Retrieved April 11, 2013, from <http://www.zakon.org/robert/internet/timeline/>
- Zeller, T., Jr. (2005, February 1). Law barring junk e-mail allows a flood instead. *New York Times*, p. A1.

This page intentionally left blank

The World Wide Web

The Web has become equivalent to the Internet in the minds of many users, but they are not synonymous terms. The Web is part of the Internet, but not everything on the Internet is part of the Web. The Web uses URLs and the Hypertext Transfer Protocols (HTTP and HTTPS) within TCP/IP to direct the communication between client and server; these interactions combined with hypertext links functionally form a hypertext database. Because the Web was such an important Internet development, this chapter will examine the history and use of the Web along with its ongoing evolution into Web 2.0 and beyond. Further discussion of Web 2.0-related technologies and their implications for libraries are in Chapter 16.

THE DEVELOPMENT OF THE WEB

In the late 1980s, a British programmer named Tim Berners-Lee was working on a system at CERN in Switzerland to enable scientists to share and inter-link documents. Early in this process, it became apparent that, as with Gopher, the technological solution he was developing as a local information-sharing problem could be extended to the global Internet. To their credit, the people at CERN encouraged Berners-Lee in this broader vision. Berners-Lee's early work led to the establishment of all the key elements of the Web. Of course, this new information system needed a name, and Berners-Lee considered several. One early candidate name was TIM (for The Information Mine), but Berners-Lee decided this sounded a little too egocentric (Berners-Lee, 1999). The idea of interlinked resources naturally suggested a "Web," and it was also now viewed as a global possibility, so it became the World Wide Web, also known as WWW or simply, the Web.

24 Internet Technologies and Information Services

As with many great ideas, earlier visionaries, programmers, and engineers influenced it. The modern idea of creating a technological system bringing the knowledge of the world to a desktop goes back at least to 1945 and the Memex proposed by Vannevar Bush (1945) in his seminal article "As we may think." The Memex as described was a great idea but the wrong technology. The idea of hypertext had also been around for some time; most attribute the term to writer and Xanadu project originator Ted Nelson, who coined it in 1965 (Wolf, 1995). Digital hypertext databases became common by the mid-1980s in the form of CDROM databases and encyclopedias. However, Berners-Lee extended these ideas to develop a global Web of information by combining the Internet with hypertext. He also developed all the essential elements needed to make it a reality, including a transmission protocol (HTTP), a resource identifier scheme (the URI, which became the URL), and a simple markup language (HTML), which made it possible to use simple ASCII (American Standard Code for Information Interchange) text programs to create sharable, interlinked content across the globe. These technologies will occupy significant parts of this text. The Web was first publicly demonstrated during the Hypertext '91 Conference in San Antonio Texas, and the Internet has not been the same since then (Berners-Lee, 1999). In celebration of the 20th anniversary of the Web in April of 2013, CERN launched a project to preserve the first Web page, shown in Figure 2.1.

The first view of the Web required a direct telnet connection to CERN. There were just three websites available on this embryonic Web, so the idea of

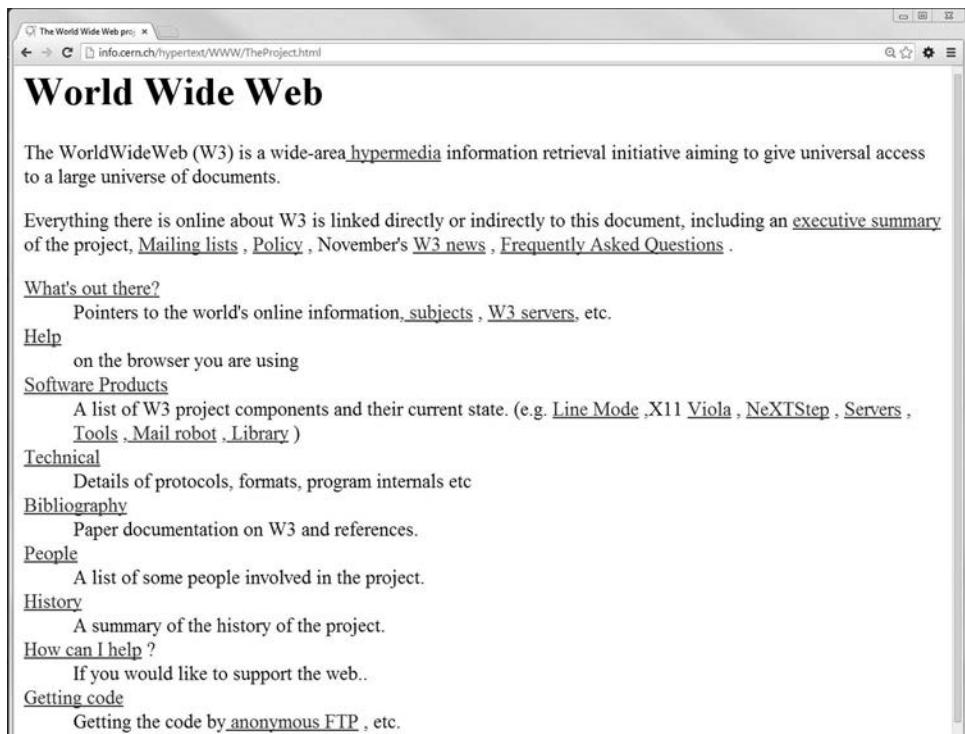


Figure 2.1 The Web page at CERN (used with permission).

browsing to find content was a reasonable retrieval strategy. The first personal Web browser clients were text-only and designed for the UNIX platform, such as the Lynx browser developed at the University of Kansas. Anyone with a shell account on a UNIX system where this browser had been installed could view Web content. However, the release of the Mosaic browser brought the Web into the public consciousness. Mosaic was developed by a team led by Marc Andreessen at the NCSA at the University of Illinois Urbana-Champaign. Andreessen went on to form Netscape Corporation, which became the dominant browser client through much of the 1990s until Microsoft entered the browser market with full force. The Internet Explorer (IE) browser soon gained a dominant market share during this period, but the strategy of bundling it with the Windows OS resulted in an unfair practices court challenge. When the dust settled, the browser wars ended with IE still in a dominant position, but other browsers such as Netscape, Opera, Firefox, Safari, and now Google Chrome have a significant market share.

KEY WEB TECHNOLOGIES: URLs, HTTP, AND HTML

The key elements to the development of the Web included an Internet protocol to specify the client-server interaction, a way to identify the location of a resource, and a way to create content not dependent on a proprietary format, which was the HTML format. These technologies of the Web are explored further in Chapters 4, 5, and 7. Initially, Berners-Lee (1999) proposed a URI, a Uniform Resource Identifier (URI), as the way to identify a resource, but the IETF standards body changed this to the now familiar URL, the *Uniform Resource Locator*. URLs have a standard syntax. They begin with a protocol followed by a colon and a double forward slash, followed by a *Fully Qualified Domain Name* (FQDN, i.e., a host at some domain), followed by the directory path to a resource ending with the full file name. Figure 2.2 shows the structure of a sample standard URL.

Besides the usual HTTP that start the standard URL, they can also accommodate the other Internet protocols that were prevalent at the time of its development, for instance a URL could designate “gopher” or “TN3270” as the protocol in place of the HTTP. So in addition to `http://hostname`, “`tn3270://hostname`” or “`gopher://hostname`” could be valid URLs as well. Filenames are

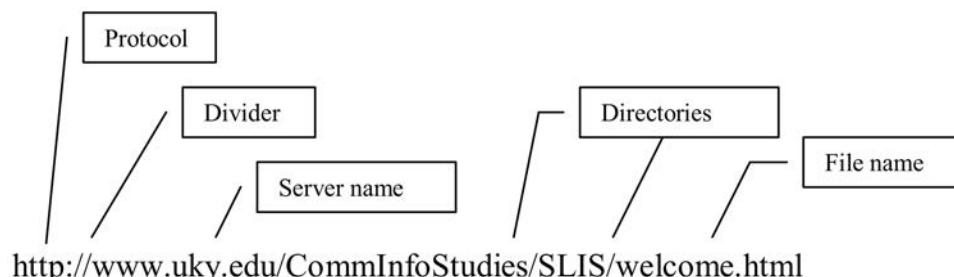


Figure 2.2 The format of a standard URL.

not always required elements to end a URL because several default names are sought by a server when the path ends with a machine or directory name; “**welcome.html**” or “**index.html**” is sought by the server without being specified in the URL. URLs frequently point to resources on Web servers running on UNIX machines, making case sensitivity a potential issue; however, most browsers have a “smart browsing” feature to overcome this problem. Long URLs can be shortened with the creation of an alias name and a subsequent HTTP redirect; for instance, in the URL shown in Figure 2.2, the **CommInfoStudies** portion can also be replaced with the abbreviation “**CIS**.” Alternatively, services such as TinyURL.com can be employed to provide shortened URL aliases and redirects; link resolver services are described further in Chapters 7 and 16. However, when URLs appear in HTML the match must be exact; a link reference within HTML href code that did not use the mixed case in the **CommInfoStudies** portion of the reference could result in a “page not found” error in some browsers. The Web developed within the TCP/IP framework, which is covered in Chapters 4 and 5. Although much of the discussion of Web protocols will be deferred to Chapter 5, it is important to note here that the Web required the development of a new protocol—the *hypertext transfer protocol* (HTTP) along with a more secure version designated with HTTPS. This protocol controls the client server communication between the browser and the server, the Hypertext Transfer Protocol Daemon (HTTPd). The protocol defines how the client forms a request header that is responded to by the server. HTTP was developed as a “stateless” protocol and it required individual client–server interactions for each piece of content associated with a page. For example, if a Web page has three image tags four separate interactions between the client and server take place: one for the HTML file and one for each image. The lack of any record of each client–server connection created issues for transactions where an ongoing connection was needed; these issues were dealt with by *cookies* and subsequent enhancements to the protocol itself. As part of the TCP/IP family, the protocols of the Web are described further in Chapter 5.

The other essential technology required for the Web was a content format that would be text based and that accommodated the essential linking function that is fundamental to its use. The development of the Hypertext Markup Language (HTML) accomplished this with a simple ASCII text format that used text for both the information and for the tags that control its presentation. The idea of this approach was not new nor was the idea of “markup” languages—HTML was derived from an existing standard, the *Standard Generalized Markup Language* (SGML). The key idea of the Web is linking, and this is accomplished with the anchored hypertext reference tag. HTML is one of the foundational technologies of the Web and Chapter 8 is devoted to exploring it in detail. Understanding HTML is essential to many parts of this text: it is important as the underlying technology of Web design (Chapter 7), as a model for XML (Chapter 12), as an Internet content format (Chapter 13), and as the data source for link analysis ranking algorithms (Chapter 15).

THE INVISIBLE WEB: BELOW THE SURFACE

The invisible (aka the “deep” or “hidden”) Web is associated with content that cannot be found by standard search engines, making it generally unavailable

to most users. The invisible Web represents a potentially huge source of important content that some estimate could be many orders of magnitude greater than that of the open or “surface” Web and it is growing faster than the visible Web. It is estimated that for every “findable” URL in a search engine there are likely 5–10 in the hidden Web (Devine & Egger-Sider, 2009). Content can be “invisible” to search engines for many reasons—the content might be protected by a firewall or authentication requirement; reside on a server that has implemented the robot exclusion protocol at either the site or document level; be an unknown file format; be dynamically generated; or reside in a private proprietary database.

The boundaries between the visible, searchable Web and the hidden Web are blurring as some search engines can access resources that might be invisible to another, such as with content in Portable Document Formats (PDFs). Although the reach of large-scale search engines is impressive, invisible Web content is important both because of its vast amount and because it is often of high quality. Librarians are acutely aware of the public perception that everything they need is available via a Google search. This is a pertinent issue for libraries because many of the high-quality Web resources they offer are not freely available to public search engines and are therefore part of the invisible Web. As explored in Chapter 16, the perception that everything should be available through a simple interface such as Google’s has influenced user expectations for library systems. Libraries are seeking ways to offer simplified federated search options with similar interface simplicity. In addition to accessing invisible Web content through a library portal metasearch, there are also a number of specialized search engines that can help identify and retrieve invisible Web content. A good source for these is by Scheeren (2012).

WEB 2.0

The term *Web 2.0* coined by Tim O’Reilly of O’Reilly Publishing has come into general use to describe a transition from the initial Web, consisting primarily of content housed in static HTML pages residing on isolated servers (i.e., Web 1.0), to a more dynamic and interactive computing platform supported by a number of new technologies and protocols. Web 2.0 is characterized by applications used for collaboration (wikis), interactivity (blogs), social networking (MySpace, Facebook, LinkedIn, etc.), and mashups (various forms of Web application hybridization resulting in new functionalities). Protocols and technologies supporting or enabling Web 2.0 include RSS, Simple Object Access Protocol (SOAP), Asynchronous JavaScript/XML (AJAX), and Representational State Transfer (REST), as well as Cascading Style Sheets (CSS) and scripting for database-driven dynamic content, some of which are addressed in other parts of this text. Web 2.0, the future of the Web, and its uses in libraries are all discussed further in Chapter 16.

THE MOBILE WEB

There is only one Web, but it can be accessed and utilized in many ways. The “mobile Web” simply reflects the reality that more and more people are

accessing the Web with a smartphone or tablet device over a Wi-Fi or cellular data connection. This shift to mobile is one of the broad megatrends discussed in Chapter 1. Smartphone shipments overtook those of the PC in 2010 (Goldman, 2011). This is not too surprising as the cost of smartphones or tablets became reasonable and connections got faster many people who used a PC primarily just for Internet access found a mobile device more convenient and useful for that access. This trend to mobile has huge implications for both consumers and for organizations wishing to reach their mobile clients. It also has ramifications for Web designers and developers, because mobile access creates specific challenges for site design, information delivery, bandwidth utilization, and accessibility, which are explored in other parts of this text. There are a few stereotypical characteristics of mobile users such as the hurried executive or the busy student, but the reality is that we are all becoming mobile users. Devices are not really “mobile”—the people using them are. Sometimes the mobile user is the person “on the go” but mobile devices are also becoming the most common Internet access preference even when we are just relaxing at home.

THE SOCIAL WEB

Web 2.0 has been defined as a platform, and that raises the question, “A platform for what?” For many, it is a platform for collaboration and using social media. The Internet has supported collaboration and social interactions from its inception—email distribution lists, FTP of shared files, Usenet groups, personal Web pages, and even the earlier Bulletin Board Systems (BBS) all fostered sharing of information and connecting to other people. However, the scale and depth of these interactions have grown exponentially in the last decade—a recent IBM commercial refers to the 3.2 billion comments and posts made per day in various social media to make the claim that its use is essential to success in the marketplace. The growth of social media has been driven by both the proliferation of sites that reach and engage new users and by the growing dominance of the mobile experience for these activities. Mobile access accounted for 63 percent of the growth in the overall time spent using social media in 2012. In addition, social media is turning traditional media such as TV into a shared social experience—according to The Nielsen Company as of mid-2012, 33 percent of Twitter users in the United States tweet about TV-related content and these numbers range from 50 to 60 percent in Latin America and the Middle East (State of the media, 2012).

Because the tools of social media are associated with Web 2.0, they are often used interchangeably, but social media is not just about the tools. It is driven by willing participants who create and monitor user-generated content (UGC). Kaplan and Haenlein provide a useful definition of social media as “a group of Internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of user-generated content” (2010, p. 61). The power of social media is in the social capital it accrues for an organization, and further consideration of its application in libraries continues in Chapter 16.

Blogs and RSS

Blogs were an early social media tool that quickly gained acceptance within a few years of their introduction and have become important both to the participants and as a source of searchable content. The name is derived from the term “Web log.” Blogs are a Web-published narrative of an individual or group who syndicates the content to subscribers and creates a shared and regularly updated source of commentary. The blogger frames the discussion, initiates the threads, and decides if and how comments can be added. Some blogs are very personal narratives, and others focus on a particular subject area. The software to create and manage blogs can be managed on a server the blogger controls or placed on a Web-hosting service such as Bloglines (<http://www.bloglines.com/>), Blogger.com (<http://www.blogger.com/start>), or Wordpress (<http://www.wordpress.org>). Local reader software or a reader website is used for viewing the syndicated content of RSS feeds. The popular Twitter service is a form of microblogging where the posts are “tweets” that are limited to 140 characters and topic threads on different themes identified by hashtags (#term).

Wikis

The Wiki was developed by Ward Cunningham (Bishop, 2004). The name is from the Hawaiian for “quick” and refers to software enabling anyone to add or edit Web content without having to know HTML or any details about the hosting platform. Wikis have not only become an important source of Internet-based reference, such as the online encyclopedia wikipedia.com, but they also present opportunities to build and share many forms of content. Wikis can be used to “crowd source” any type of question or problem to build a shared knowledge base. Some examples relevant to libraries are discussed in Chapter 16.

Social Networking Sites

The idea of social networking sites (SNS) goes back to early virtual communities that appeared in the mid-1990s such as Geocities; most users today associate them with sites such as Facebook, MySpace, LinkedIn, Google Hangouts, and Twitter. However, the term “social” can be applied not only to the sites that promote the idea of friendship but also to any site that depends on broad communities of engaged participants such as Wikipedia and Pinterest. MySpace, once the largest such site, relaunched itself in 2013 with a redesigned site, mobile app, and a music streaming service, but it is unclear if it can overtake the dominant Facebook (Stenve, 2013). Facebook is known for extending the notion of “friendship” and offers many tools to support their online community with the posting of profiles, photos, messaging, and chat. As a testament to its success, between 2008 and 2011 the number of those using social media doubled and there are more than a billion active Facebook users (Hampton, 2011; Tam, 2013). Whether it will remain the dominant SNS is an open question—a 2014 study suggests Facebook might follow the boom and bust cycle of other social media sites, predicting it could lose nearly 80 percent

30 Internet Technologies and Information Services

of its users between 2015 and 2017 (Cannarella & Spechler, 2014). LinkedIn is a similar site with a business networking focus that makes it popular with many professionals. Other important social networks along with their respective percentage of online adult users include Twitter (16%), Pinterest (15%), Instagram (13%), and Tumblr (6%; Brenner, 2013). The social Web is a rapidly changing world and newer companies, such as Reddit, a source for what is new and popular on the Web, and Snapchat, a photo messaging application, are quickly gaining attention.

Concurrent with the growth of social media during this period was a shift in the demographics of this audience from being primarily younger early adopters of these services to include older adults with the average age of Facebook users increasing from 33 to 38 years during that period. Not only do social media appear to be growing dramatically, but also they are reaching a wider range of the population. It is not surprising that most businesses now include social media in their business plan, not just for advertising and promotion but also for customer support and reviews.

VIRTUALIZATION, GRIDS, AND CLOUDS

These three interrelated topics are the subject of much interest today and are the subject of an excellent summary book *Grids, Clouds, and Virtualization* (Carfaro & Aloisio, 2010). The “cloud” has become a ubiquitous reference used in many different contexts but in some ways, many of the key ideas of each have been around since the beginnings of the Internet—an important initial goal of the NSFNET was to distribute and share access to regional supercomputer centers. The idea of “local clouds” is a direct successor of the thin-client/Java model that Sun Microsystems promoted in the mid-1990s, but local clouds now often might mean accessing a home, office, or enterprise storage device such as Western Digital’s “My Cloud.” The idea of a grid goes back to the beginning days of electrification in the early twentieth century. Creating a “virtual machine” (VM) that carved out a portion of the systems resources to give the appearance of a dedicated single user machine became possible early on in the history of mainframe and UNIX host computers. These three technology concepts have converged with the Web and have many important implications for libraries and information organizations, especially as we are becoming increasingly dependent on mobile devices for computing and information seeking.

Virtual Machines and Virtualization

Virtualization allows for the decoupling of hardware from software. A *virtual machine* (VM) refers to how the computing resources of a single physical machine can be partitioned and allocated to multiple users all running what appear to the user to be multiple separate machines. Each instance is then a VM that partially or fully emulates the architecture of a real-world computer. This concept goes back to the 1960s and IBM mainframe VM-operating system (OS) that gave multiple users access to a dedicated portion of system hardware

resources and a “copy” of an operating environment such as the Conversational Monitor System that then could run applications that were all protected from each other and other users. The VM idea has been extended to all forms of computing, from UNIX hosts supporting multiple users, to Apple computers running a virtual Windows machine or Windows computers running a virtual DOS session. Various forms of virtualization are the foundation for much of the cloud computing possible today.

Although the idea is not new, the connectivity of the Internet has greatly increased the utility and the role of VMs and virtualization services. Virtualization can be implemented in different ways and on various levels, but it usually takes place in one of two forms, depending on whether or not multiple OSs must be supported on the hardware node. The first strategy is the creation of virtual containers that run the same OS. In this scenario, the hardware hosting the VMs has the core part of that OS already installed, which can then run multiple VM instances of that OS. That preinstalled piece is called the kernel, and it communicates with the hardware and handles system calls for each VM instance. In this case, each user’s VM is only a partial installation of the OS and all the VMs created are dependent on that common kernel. This is a very efficient strategy because hundreds of such VMs can be supported on one hardware platform at a relatively low cost per container. The other strategy uses a software layer called the *hypervisor* that provides a platform on which multiple OS stacks and associated applications can be deployed (Csaplar, 2013). Each of these VMs has a full installation of the desired OS.

Virtualization has quickly become a widely adopted IT solution that presents many advantages and potential cost savings. The three most typical forms of virtualization are implementations of virtual servers, virtual storage solutions, and virtual clients. Server virtualization offers several advantages including the relative ease and speed with which new servers can be deployed as well as direct cost savings resulting from fewer physical servers to purchase and maintain. Instead of buying and maintaining dedicated server hardware to support Web development, many institutions are encouraging the use of a VM on a centralized system. Such a solution not only has the initial cost savings of purchasing your own hardware, but a centralized VM host is also usually in a secure and climate controlled environment with appropriate firewall protection and power backup. The downside is that the user sometimes has to know how to build the VM from the ground up unless the IT department provides that service. This involves installing the preferred OS and server applications such as those found within the LAMP package of Linux, Apache, MySQL, and PHP. Once the OS and needed developer tools are installed the VM behaves identically to a standalone Linux computer dedicated to that user. For the public and many businesses, Amazon has become a major vendor of VMs through their Amazon Elastic Compute Cloud (Amazon EC2, <http://aws.amazon.com/ec2/>) services. Machine instances are available to support a wide range of Web and computing activities with a flexible array of OSs and platforms available in a controlled and secure environment. Pricing and capacity are also flexible with on-demand instances and hourly pricing.

Storage virtualization allows all storage devices to be controlled by a single management application, providing smoother and more integrated data movement through a storage area network (SAN). Desktop or mobile client

32 Internet Technologies and Information Services

virtualization reduces the device's role to that of capturing input from the keyboard, mouse, or touch screen and then painting the screen on the display. There may not be as many hardware cost savings associated with desktop virtualization, but it can extend the working life of older PCs and OSs that would otherwise have to be replaced. Advantages include savings for technical support and application deployment as well as better data security. As with server virtualization, client software is needed to provide the connection and mediate communication among the remote host and the local OS and hardware. Examples of such desktop software are the Microsoft Virtual Desktop Infrastructure (VDI), Citrix XenDesktop, and VMware View. Virtualization of the mobile experience allows people to connect to their work from any place/any time and virtually run applications on devices with limited processing power and storage. Such virtualization also is a valuable way to leverage the many personal mobile devices that people already have to be able to perform work-related activities, and many companies have embraced the BYOD (Bring Your Own Device) model.

There are many examples of client virtualizations, such as the apps that support a remote desktop connection to a PC that has been configured to accept remote logins. There are also services designed to deliver a Windows experience to desktop and mobile users. For example, the free Onlive Desktop (<https://desktop.onlive.com/>) service turns any tablet into a Windows desktop with the appropriate iOS or Android app. Once logged in, the tablet presents a standard Windows cloud-based desktop with MS Word, Excel, PowerPoint, Flash, and Gmail service; files can be created and stored on 2 GB of free storage (fee-based services include additional applications and storage). Another example is a service offered by my university that allows students to access a virtual Windows desktop through the Web, providing remote access to a set of standard campus-licensed applications. After installing the needed Citrix receiver client, students can connect to a virtual Windows desktop from any platform or device using their standard network credentials. Branch libraries on our campus have utilized this virtual system, thereby reducing staff time previously needed to manage and secure multiple Windows computers for patron use. There are numerous benefits to these solutions, but performance can sometimes be problematic with many users. In addition, users must know how to install and use the client software to enable the connection to the remote host.

Virtualization can be at the application level such as with the *Java Virtual Machine* (JVM) that can run Java byte code. Sun Microsystems introduced Java and the JVM in the mid-1990s, initially to support the thin-client model architecture, but Java has become essential for many Web applications and is now installed on over 4 billion desktops and phones (Learn about Java technology, 2013). The JVM machine can be run by any OS making Java applications platform independent. The Java Runtime Environment (JRE) is a plug-in that allows applets to run in various browsers and supports many Web 2.0 applications (What is Java, 2013).

Grids

In *The Big Switch*, Carr (2008) describes the early history of electric power generation that required each town or factory to invest in the equipment and

expertise necessary to create the power they needed. The Edison Company's business model was to manufacture and sell this equipment and expertise repeatedly. However, a competing strategy soon emerged that instead was focused on centralizing power generation and distributing it where needed via a power grid that could manage the load and dynamically respond to variation in demand. The efficiencies gained by the consumer were quickly apparent and resulted in a shift that profoundly changed the industry and society.

A similar paradigm shift is happening in the world of computing. Early definitions of grid computing centered on reliable, consistent access to inexpensive high-end computing. More recently, this description has evolved to focus on an infrastructure for sharing resources in a dynamic, noncentralized fashion by utilizing standard protocols (Carfaro, 2010). Although this sounds much like cloud computing, those technologies are typically controlled centrally by the service provider. Grid technologies are a distributed model where multiple hosts work in conjunction to manage the computing load. Each computer in the grid can share resources with any other computer in the grid. Together, the nodes of a grid can act in unison on a computing problem and behave like a single, albeit virtual, supercomputer. Two examples of grid computing are the SETI@home initiative that allowed any Internet-connected computer to run a shared program to analyze radio telescope data and the NSF TeraGrid project that integrated distributed access to high performance computing.

The simplest type of grid is a cluster of computers with a single owner run by a single organization. At the other end of the spectrum are global grids with multiple owners running across multiple sites and organizations. Regardless of their scale, the commonality is that to the user, the grid is seen as a single computational resource. Each computer on the grid is a node and each must be running a type of networking software with the protocols needed to allow it to function within the grid and work with other computers with different local OSs. Software is also needed to manage and administer the grid, a task usually assigned to a grid server, often referred to as the control node or master host. Middleware refers to the set of programs that enable the machine-to-machine (M2M) communication needed to integrate the activities of the computers in the grid.

Cloud Computing

Cloud computing has come to mean many different things in different contexts. The notion of cloud computing does seem inexorably linked to the Web 2.0 view of the web as a platform and the use of a browser to interact with many applications and data services. In this model, the browser takes on an extended role, running cloud applications much as a local OS does on the PC. The main characteristic of cloud computing is based on the idea of moving away from local computing processing or data storage capabilities to Web-based solutions through services delivered to users over the Internet using the provider's infrastructure. Services range from simple data storage with Google Drive, Dropbox, Apple's iCloud, or Microsoft's OneDrive, to running application services such as Google apps, Zoho, and Prezi, or leasing a full platform such as Amazon's EC2. A number of cloud services for storage and applications are described later in this chapter.

Cloud Architecture

Cloud computing can take place locally within an organization, globally via the open Web, or utilize a hybrid approach. Cloud computing options are available to provide service solutions that can take the place of local infrastructure, applications, and storage. These include Infrastructure as a Service (IaaS), such as VMs and storage devices, Software as a Service (SaaS), such as productivity applications or email, or Networks as a Service (NaaS), such as with a VPN. Cloud computing is similar to client-server architecture except the two pieces of the interaction are not a neat duality with specific well-defined roles but instead are a collection of various front-end and back-end technologies. The back end of the cloud is comprised of the various servers and storage devices needed to authenticate users, deliver applications, and store user data and preferences. The back end may be as simple as a single server or as complex as a grid of many computers and devices. As with a grid, a central server usually manages the cloud users and applications, and that server may itself be partitioned into multiple virtual servers each handling different functions. The front end is what we use to connect to, and work within, the cloud; often this front end is as simple as a standard or enhanced Web browser. As with client-server approaches, there is a division of labor with each component handling some of the processing needed, but often most of the processing and storage are managed by cloud resources. One advantage of cloud approaches then is that the user's computer or device just needs enough memory and processing power to handle the demands of the front-end application in use, and there is little or no need for local storage of data. Devices designed for the cloud such as the Chromebook have local storage, but purchase of the device also includes free storage on the cloud with Google Drive. It is evident that there is significant overlap among these interrelated models and technologies. Cafaro (2010, p. 8), summarizing a few of the interconnections that emerged from the 2008 Open Grid Forum, notes that although clouds may be thought of as "virtual grids" of "VMs," they are different, because unlike grids, clouds rely on a simple, high-level interface. Clouds can be built from grids, grids can be built within the cloud, and virtualization is key to how clouds and grids can be used for remote computing via the Web.

Grids and clouds may become the new computing paradigm, but they present security, privacy, performance, and reliability challenges. Unexpected loads can disrupt the power distribution grid, and when combined with a failure at a key location, a local disruption can cascade into a massive outage as happened in the Northeast United States in 2003. Although such failures are uncommon, they do occur and similar problems can affect grid and cloud computing. Just as with the power grid, cloud and grid computing requires the efficient matching of supply and demand by creating a scalable computing infrastructure. Traditionally dealing with spikes in demand required providing extra capacity to ensure it would be available when needed, but newer approaches are based on the idea of resource elasticity that allows for both the expansion and contraction of available resources according to the demand at a given time (MacVittie, 2013).

Cloud Application Examples

There are hundreds of cloud applications and services available today and there are many sources that list useful cloud applications (Yonatan, 2013). The list below is intended to highlight a few important services within each category:

- Productivity applications: Microsoft Windows applications such as the Office suite can be accessed with the Onlive Desktop highlighted earlier. Microsoft's Office 365 (<http://office.microsoft.com/en-us/>) is a fee-based cloud service that provides access to Microsoft products and utilizes OneDrive for data storage. Zoho.com is a comprehensive collection of Web customer relationship software programs for business that includes collaboration tools and productivity apps. Prezi (prezi.com) is a frequent PowerPoint substitute that enables presentation sharing anywhere as well as real-time collaboration. Google Drive provides access to productivity apps for creating or editing documents, spreadsheets, forms, and pictures as well as access to hundreds of other apps for other activities. However, when uploading Microsoft Office files one must decide if files are to be converted to Google's formats or left in the native format; conversion of the file can result in the loss of some features in the original file, which can be an issue for group work.
- Storage solutions: As mentioned earlier, Amazon offers cloud services that include mass storage. Google Drive was an early and very successful storage site tied to Google applications and an account offers 5 GB of free storage. As with many of storage services, Google Drive can be synchronized with your desktop or mobile device. Apple's iCloud also offers 5 GB of free storage as well as synchronization with, and streaming from, iTunes. Microsoft's OneDrive provides 7 GB for free. It is optimized to work with the Windows 8 file manager and IE and allows both synchronization and remote access to files on any device associated with your Microsoft account. Dropbox is also a very popular storage solution with 2 GB of free storage. Unlike some of the other options, it does not need to be accessed via a browser and can be used with any OS's file management tools where it behaves as any other network drive.
- Photo and video sharing and management sites: These sites are not just for storage—many storage solutions offer both image viewers and minimal editing capabilities for these file types. Social media sites such as Facebook, Instagram, and Snapchat have become popular ways to share photographs but those with large numbers of images or who desire additional tools to edit their work often turn to dedicated photo sites that typically include many more powerful and useful features and tools. Many have editors that allow photos to be enhanced and organized, making it easy to share photo albums or incorporate photo feeds with a public API (Application Programming Interface). Flickr, owned by Yahoo, offers a free terabyte of storage and a wide

range of editing tools. Photobucket is another option that has many social networking features, but some find the site too ad intensive. Google's Picasa photo sharing service has added new tools that make it comparable to Flickr. For video, Google's YouTube is the most common location to store and share videos and with 72 hours of video being uploaded per minute, it is clearly the dominant choice. However, there are alternatives such as Vimeo (vimeo.com), perceived by many videographers as a more professional option (Larson, 2013). Eyejot (<http://corp.eyejot.com/>) is an easy way to send a quick video message from either a desktop or a mobile device.

- Meetings/communication: Skype (<http://www.skype.com>), now owned by Microsoft, has been a long-time cloud solution for two-way video chat that also supports instant messaging (IM) and standard voice calls; premium versions support group video conferencing. Lync is Microsoft's communication service for video conferencing and IM, and the cloud version called Lync Online (<http://www.mylynconline.com/>) is one of the Microsoft 365 products. For professional-looking mass mailings and newsletters, MailChimp (<http://mailchimp.com/>) has a number of features for creating the message, managing lists of recipients, and automating the sending of follow-up messages.
- Getting user input and arranging meetings can also be made simple in the cloud. Despite its frivolous sounding name, SurveyMonkey (<https://www.surveymonkey.com/>) makes creating, distributing, and analyzing surveys very easy. For real-time, live polling, try Poll Everywhere (<http://www.polleverywhere.com/>). To arrange meetings, Doodle (<http://www.doodle.com/>) makes it simple to offer various meeting times and quickly determine participant availability.
- Special interest and library related: Library Thing (<http://www.librarything.com/>) combines cataloging, reviews, and social networking to form what it describes as the "world's largest book club" with over 1.7 million users. Oysterbooks (<https://www.oysterbooks.com/>) hopes to replicate the Netflix model for readers, offering unlimited access to eBooks for a small monthly fee.

Extending the Browser

As mentioned earlier, the browser is often the front end to using cloud applications. Browsers have evolved far past their initial simple Web-browsing role, behaving more like an OS capable of running cloud-based applications. All modern browsers are extensible and each has a huge array of plugins and tools available that can add new functionalities, enabling the browser to act as the platform for many cloud applications such as calendars, word processing, image viewers, email, ad blockers, and JavaScript debuggers. The browser has a central OS role in the Chromium OS for the cloud-based Chromebook, where the only other native apps are a file manager and a media player.

There are important browser extensions that can enhance accessibility as well as many tools for Web developers. Browser tools that might be useful

for Web developers using Chrome include CSS viewer, JQuery code checker, IP address and domain checker, XML Tree, cookie edit, JavaScript console, and various analytic tools, to name just a few. The Firefox Web developer toolbar provides quick menu access to view or disable CSS, JavaScripts, HTML source code, cookies, and validation tools. The downside of the evolving browser environment is that it can become a challenge for developers to stay abreast of frequent browser version changes. For example, the Blackboard Learning Management System (LMS) updates behave differently when used with different browsers or with browser version updates. Sometimes the only solution is to use an earlier version of the browser or a compatibility mode if it is available. For example, IE's compatibility mode, activated by clicking on the compatibility view button in the address bar, allows the emulation of an earlier version of IE for sites that do not work correctly with the latest version.

USING THE WEB

The Web was an immediate success, growing faster than any previous mass-communication technology. It took radio 38 years to reach 50 million listeners and 13 years for television to reach that number of viewers. The Web reached that many users in North America in less than five years. Whether with a PC or a mobile device, consumers are spending more time than ever on the Internet; according to the Nielsen Company overall usage was up 21 percent between July 2011 and July 2012 (State of the media, 2012). The rapid acceptance by the public of a technology that required a significant investment in computing equipment, connection costs, and in developing needed IT skills is a compelling testament to the usefulness and the value of the Internet. This section highlights its importance to information seeking, commerce, and education.

Information Seeking

The Web has dramatically changed how people look for information, which in turn has affected libraries and librarians. The ease of finding useful information is primarily due to the development of free and powerful search engines. Search engine development throughout the short history of the Web has created a reinforcing cycle that drives the ever-increasing number of resources available and the concurrent rise in the number of people turning to it for information seeking, shopping, and entertainment. How the Web has influenced information behavior and the technologies of search engines are especially relevant and comprise much of Part 3 of this text.

eCommerce

One measure of the success of the Internet is Web commerce. According to the U.S. Census (2008), the \$127 billion Americans spent in 2007 on e-commerce rose to about \$1.4 trillion by 2010, a 10-fold increase in just three years, representing about 16 percent of the entire economy. Amazon, once just an

online bookseller, now offers one-stop shopping for most consumer items. As noted earlier, sectors such as print media and the broadcast, music, and video industries have been particularly affected by the shift to Web-delivered media. Services such as iTunes, Pandora, and Spotify have captured large audiences who no longer buy music on compact disks. Streaming services such as Netflix, Hulu, and Amazon Prime provide entertainment options that no longer require people to have cable or to use broadcast television.

Mobile Web technologies have had a tremendous impact on ecommerce. Mobile devices and smartphones have not only boosted online purchasing, but they have also affected in-store purchasing behavior. A recent survey found 58 percent of cell phone owners either called someone for shopping advice, looked up a review, or looked up prices at other locations (Smith, 2013). Social media have become intertwined with all forms of commerce providing advertising, customer support, and crowd-sourced product reviews. Cloud computing and virtualization have made it easier and more cost effective for entrepreneurs to start a business using leased hardware and software. All these trends also apply to the operational assumptions of libraries and other information businesses as they continue to compete in the information marketplace.

The Web and Traditional Media

The Web has become the main source of news and entertainment for many, and it has been a two-edged sword for traditional media such as newspapers, magazines, television, and music. Although traditional media have benefitted from the use of the Web and social media, the downside has been apparent for some time by a number of measures. Television network market share fell 33 percent over the 20 years ending in 2005, and music CD sales were down 21 percent from their peak in 1999 (Nielsen Media Research in Storey, 2005). Even though there has been a resurgence of interest in vinyl records with sales rising 14 percent between 2006 and 2007, CD sales have continued to decline, falling to 360.6 million in 2008 from sales of 553.4 million in 2006. Digital music has continued to grow during that period with MP3 sales up from 32.6 million to 65.8 million (Mearian, 2009). For video, retail stores like Blockbuster have gone away while Netflix streaming output topped 1 billion hours a month in 2012 (King, 2012).

Newspaper circulations have also declined, while Internet use has grown; starting in 1994, overall newspaper circulation fell by 7.9 percent during the first decade of the Web (Sloan, 2006). Further evidence of the stress on newspapers is the steep decline in advertising revenue, which fell to a 60-year low in 2011 (Perry, 2012). The decline has steepened in the last decade; the Pew Research Center Project for Excellence in Journalism reported revenues in 2011 were one-half that of 2007 and one-third of 2001 amounts (The state of the news media, 2013). Magazines are experiencing a similar fate, with popular titles such as *Newsweek* and *SmartMoney* dropping print in favor of digital-only versions. It is not surprising that many traditional media are scrambling to find new revenue from online advertising, special online features for digital subscribers, niche marketing of services such as in-sourcing of printing capacity for custom publishing, and embracing the contributions of bloggers and

“citizen journalists” to include amateurs in the content stream (Doctor, 2012; Ingram, 2012). Even with these adaptations, some major papers are predicted to fold in the near future (McIntyre, 2009).

eBooks represent another shift in traditional media use that is of particular interest to libraries. By the end of 2012, eBooks accounted for 22 percent of all book spending, up from 14 percent the previous year; Amazon reportedly has 27 percent of that total (Milliot, 2012). Publishers continue to impose restrictions on eBooks that present access challenges to libraries and their users, such as stringent Digital Rights Management requirements. There are also format and device differences that further complicate library lending of eBooks to their users. However, there are huge repositories of digital works in the public domain that are available through Project Gutenberg (<http://www.gutenberg.org/>), Feedbook (www.feedbooks.com/publicdomain), and Google Books. Just as Netflix has changed the video marketplace, eBooks and digital magazines are changing the library and publishing world; Oyster (<https://www.oysterbooks.com/>) offers unlimited access to over 100,000 eBooks for a monthly fee and Zinio (<http://www.zinio.com>) offers digital magazines.

The effects of the Long Tail discussed in Chapter 1 are especially evident with the rise of many digital businesses such as Amazon, Netflix, eBay, and iTunes that offer extensive libraries of often-obscure and hard-to-find content. The fate and future of traditional media are relevant to libraries and other information businesses as they continue to move from the print-centered past and seek to compete in the changing media marketplace.

Education

Since its inception, the Internet has been promoted as a way to extend and expand learning both in and out of the classroom. There have been a number of critiques of over dependence on technology in education, from the early critique of Stoll (1995) in “Silicon Snake Oil” to current concerns about the distribution of tablets to elementary students (Iasevoli, 2013). However, most educators see value not just in the huge number of information resources the Web offers but also in the collaborative tools and apps that allow for new pedagogies. In higher education, Internet-delivered distance learning with fully online or hybrid courses are bringing educational opportunities to many who are not able to follow the traditional on campus or full-time student path. The program I am associated with at the University of Kentucky School of Library and Information Sciences offers a fully online master’s degree option. I have taught IT courses via online distance learning for a decade and although it presents unique challenges to both the instructor and the student, it is an opportunity to rethink the learning experience. We currently use the Blackboard course management system to deliver online courses, but a number of other Web technologies are also employed, including video conferencing with Adobe Connect, and iTunes U content such as podcasts, and videos either embedded into Blackboard or presented via YouTube. Collaborative group work is managed through discussion forums, blogs, and wikis that are part of Blackboard or used on the open Web.

40 Internet Technologies and Information Services

An exciting but sometimes controversial development is the *Massive Open Online Course* (MOOC) that are being offered by many well-known institutions. There are more than a dozen educational initiatives promoting hundreds of MOOCs, including Canvas.net, Class2Go, Coursera, edX, and OpenLearning, to name just a few. Coursera has generated a great deal of interest with 70 educational partners offering 374 courses to over 3.5 million participants (Take the world's best courses, 2013). Many universities are assessing how to respond to this rapidly changing environment; some are considering integrating MOOC courses into their curriculum and awarding credit toward their degrees, while others worry such moves will replace faculty, destroy departments, and diminish the educational experience (Kolowich, 2013).

SUMMARY

To most users, the Internet and the Web have become synonymous, as almost all our Internet activities today are Web based. This chapter began by explaining the difference between the two, and describing how the Web was developed, what protocols support it, and how it came to dominant most all of our Internet activities. Several other forms of the "Web" are described. The "invisible" Web is defined as the content on the Web that cannot be found by a standard public search engine; content can be invisible because of its format, its location (behind a firewall, within a database, or as part of a subscription only site), or because one of the various robot exclusion protocols have been employed. The "Web 2.0" web-as-a-platform idea was explored along with the technologies and activities commonly associated with it. Web 2.0 has given rise to the mobile and social Web, and it has shifted the computing paradigm to the cloud. The Web now enables virtualization of all platforms and services. Grid and cloud computing continues to provide an ever-increasing number of useful applications that run within the browser, which has been extended with numerous plugins to function much like an OS. The discussion then turned to how the Web is changing information seeking, ecommerce, traditional media use, and education.

REFERENCES

- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.
- Bishop, T. (2004, January 26). Microsoft Notebook: Wiki pioneer planted the seed and watched it grow. Retrieved October 2007, from http://seattlepi.nwsource.com/business/158020_msftnotebook26.html.
- Brenner, Joanna. (2013, February 14). Pew Internet: Social networking. Retrieved May 20, 2013, from <http://pewinternet.org/Commentary/2012/March/Pew-Internet-Social-Networking-full-detail.aspx>.
- Bush, V. (1945). As we may think. *Atlantic Monthly*, 176, 101–108.
- Carr, N. (2008). *The Big Switch*. New York: W. W. Norton & Company.
- Cafaro, Massimo, & Aloiso, Giovanni. (2010). *Grids, clouds and virtualization* (1st ed.). New York: Springer.

- Cannarella, J., & Spechler, J.A. (2014, January 17). Epidemiological modeling of online social network dynamics. Retrieved January 22, 2014, from <http://arxiv.org/pdf/1401.4208v1.pdf>.
- Csaplar, D. (2013). Optimizing business performance: Why every C-level manager should care about virtualization. Retrieved November 11, 2013, from <http://research.aberdeen.com/1/ebooks/virtualization/download/optimizing-business-performance.pdf>.
- Devine, Jane, & Egger-Sider, Francine. (2009). *Going beyond Google: The invisible Web in learning and teaching*. New York: Neal-Schuman Publishers, Inc.
- Doctor, Ken. (2012, April 12). The newsonomics of small things. Retrieved April 16, 2013, from <http://www.niemanlab.org/2012/04/the-newsonomics-of-small-things/>
- Goldman, David. (2011, February 9). Smartphones have conquered PCs. Retrieved February 10, 2011, from http://money.cnn.com/2011/02/09/technology/smartphones_eclipse_pcs/index.htm?source=cnn_bin&hpt=Sbin.
- Hampton, Keith N., Goulet, Lauren Sessions, Rainie, Lee, & Purcell, Kristen. (2011, June 16). Social networking sites and our lives. Retrieved May 20, 2013, from <http://www.pewinternet.org/Reports/2011/~/media/Files/Reports/2011/PIP%20-%20Social%20networking%20sites%20and%20our%20lives.pdf>.
- Iasevoli, B. (2013, December 18). After bungled iPad rollout, lessons from LA put tablet technology in a time out. Retrieved January 23, 2014, from http://hechingerreport.org/content/after-bungled-ipad-rollout-lessons-from-la-put-tablet-technology-in-a-time-out_14123/
- Ingram, Mathew. (2012, April 16). The future of media=many small pieces, loosely joined. Retrieved April 16, 2013, from <http://www.businessweek.com/articles/2012-04-16/the-future-of-media-equals-many-small-pieces-loosely-jointed>.
- Kaplan, Andreas M., & Haenlein, Michael. (2010). Users of the world, unite! The challenges and opportunities of Social Media. *Business Horizons*, 53(1), 59–68. doi: <http://dx.doi.org/10.1016/j.bushor.2009.09.003>.
- King, Rachel (2012, July 5). Netflix streaming tops 1 billion hours in month for first time. Retrieved September 8, 2013, from http://news.cnet.com/8301-1023_3-57467191-93/netflix-streaming-tops-1-billion-hours-in-month-for-first-time/
- Kolowich, S. (2013, May 21). MOOC professors claim no responsibility for how courses are used. *The Chronicle of Higher Education*. Retrieved June 5, 2013, from <http://chronicle.com/blogs/wiredcampus/mooc-professors-claim-no-responsibility-for-how-courses-are-used/43881>.
- Larson, E. (2013, May 30). 5 Reasons to choose Vimeo instead of YouTube. Retrieved November 25, 2013, from <http://mashable.com/2013/05/30/vimeo-over-youtube/>
- Learn about Java technology. (2013). Retrieved May 23, 2013, from <http://www.java.com/en/about/>
- MacVittie, Lori. (2013). *ScaleN: Elastic infrastructure*. Seattle, WA: F5 Networks.
- McIntyre, Douglas A. (2009, March 9). The ten major newspapers that will fold or go digitalnext. Retrieved April 16, 2013, from <http://247wallst.com/2009/03/09/the-ten-major-newspapers-that-will-fold-or-go-digital-next/>
- Mearian, Lucas. (2009, January 2). Back to the future: Vinyl record sales double in '08, CDs down. Retrieved April 16, 2013, from http://www.computerworld.com/s/article/9124699/Back_to_the_future_Vinyl_record_sales_double_in_08_CDs_down.
- Milliot, Jim (2012, November 5). E-books market share at 22%, Amazon has 27%. Retrieved September 12, 2013, from <http://www.publishersweekly.com/pw/>

42 Internet Technologies and Information Services

- by-topic/digital/retailing/article/54609-e-books-market-share-at-22-amazon-has-27.html.
- Perry, Mark J. (2012, February 26). Newspaper ad revenues fall to 60-yr. low in 2011. Retrieved April 16, 2013, from <http://mjperry.blogspot.ca/2012/02/newspaper-ad-revenues-fall-to-50-year.html>.
- Scheeren, William O. (2012). *The hidden Web: A sourcebook*. Santa Barbara, CA: Libraries Unlimited, an imprint of ABC-CLIO, LLC.
- Sloan, S. (2006, February 19). Inklings of change—Inside the potential sale of Knight Ridder: Industry pressures, technology advances put print journalism and Herald-Leader at a crossroads. *Lexington Herald Leader*, p. A16.
- Smith, Aaron. (2013, January 31). In-store mobile commerce during the 2012 holiday shopping season. Retrieved May 22, 2013, from <http://pewinternet.org/Reports/2013/in-store-mobile-commerce.aspx>.
- State of the media: The social media report 2012. (2012). Retrieved July 7, 2013, from <http://www.nielsen.com/us/en/reports/2012/state-of-the-media-the-social-media-report-2012.html>.
- Stenovec, Timothy (2013, June 12). MySpace relaunch: Redesigned site, mobile app include MyRadio streaming music service. Retrieved August 14, 2013, from http://www.huffingtonpost.com/2013/06/12/myspace-relaunch-myradio-app_n_3428934.html.
- Stoll, Clifford. (1995). *Silicon snake oil: Second thoughts on the information highway* (1st ed.). New York: Doubleday.
- Storey, T. (2005). The long tail and libraries. *OCLC Newsletter*, 268, 6–10.
- Take the world's best courses, online, for free. (2013). Retrieved May 23, 2013, from <https://www.coursera.org/>
- Tam, Donna. (2013, January 30). Facebook by the numbers: 1.06 billion monthly active users. Retrieved May 20, 2013, from http://news.cnet.com/8301-1023_3-57566550-93/facebook-by-the-numbers-1.06-billion-monthly-active-users/
- The state of the news media 2013—An annual report on American journalism. (2013). Retrieved January 4, 2014, from <http://stateofthemedia.org/2013/overview-5/key-findings/economics-key-findings-7/#save-chart>.
- U.S. Census. (2008, May 15). Quarterly retail e-commerce sales. Retrieved May 22, 2008, from <http://www.census.gov/mrts/www/ecomm.html>.
- What is Java technology and why do I need it? (2013). Retrieved May 23, 2013, from http://www.java.com/en/download/faq/whatis_java.xml.
- Wolf, G. (1995, June). The curse of Xanadu. *Wired*, 3, 137–202.
- Yonatan, R. (2013, August 13). 35 Powerful cloud tools for modern librarians. Retrieved October 25, 2013, from <http://getvoip.com/blog/2013/08/13/35-powerful-cloud-tools-for-modern-librarians>

Network and Connection Technologies

Computer networks have facilitated the transformation of the computer from a number-crunching device into a communication device. The development of local computing networks presents many advantages and opportunities for resource sharing within an organization; these opportunities are even more compelling when extended to the global Internet. This introduction to basic network terminology, topologies, concepts, and connection technologies is necessary to understand how the Internet grew from many “interconnected networks” and to explore the various technologies employed to connect to it.

NETWORK BASICS

The term *networking* obviously precedes the computer revolution of the twentieth century, and it occurs in both technical and nontechnical contexts. There are radio and television networks, cable and satellite networks, telephone communication networks, and library consortium networks. So networking can be something done with computer hardware or with people at a professional meeting. In this chapter, networking is examined strictly in the technical context of connecting computers and other devices together for data communication and resource sharing.

Connecting computers and users together with networks facilitates resource sharing and has driven the “computing to communicating” paradigm shift responsible for the success of the modern PC; isolated PCs had limited usefulness to most people. Networking computers together has a cost, both in terms of hardware and human effort to manage and maintain them, but the payoff is the ability to share resources such as printers, disk space, data files, and applications across an organization. To accomplish this goal, three elements are

essential, represented by the “three Cs” of networks: *computers, connections, and a common language*. Each of these elements requires specialized hardware and software. The examples provided in this chapter focus on PC networks, but all computers, from minicomputers to mainframes, are likely to be part of a network. The network construct extends beyond just connecting desktop PCs to include many wirelessly connected mobile devices such as tablets or smartphones.

Networks of limited scale and geographical coverage are Local Area Networks (LANs), and multiple LANs can be connected together in a Wide-Area Network (WAN). This hierarchy is also evident in the context of the Internet; PCs connect to LANs, which can connect to WANs, which may then connect to regional networks and finally to the Internet backbone service.

THE OSI NETWORK REFERENCE MODEL

To set the stage for this discussion of networks, it is helpful to examine the conceptual framework referenced in their design. Computer and network technologies utilize designs compartmentalized into layers that can communicate with adjacent layers. The standalone PC has layers at work to manage the chain of events between the user and the hardware, as shown in Figure 3.1. People use applications, which in turn talk to the OS, which talks to the BIOS (Basic Input/Output System), which communicates with the hardware. Without these intervening layers mediating these functions, users would need to work directly with circuits and chips. When a PC is added to a network, the networking software creates other layers between users, applications, and the OS.

The developers of all computer networks confront a similar set of problems and engineers have developed a reference model to address them. The Open System Interconnect (OSI) reference model is a framework for conceptualizing the various functions a network must handle, representing them as a series of seven layers, shown in Figure 3.2. Tang and Scoggins (1992) provide extensive coverage of OSI. As the name implies, this is a reference model, not a plan for a specific network implementation. It is a way of identifying the various issues that real designs must consider, and actual designs might expand or compress some of these seven layers. The OSI model uses a similar type of hierarchy to identify the various functions the network must address, as shown in Figure 3.2.

For an Internet-connected computer there must be a TCP/IP layer above the OS. For example, Windows manages the connection between the OS and TCP/IP with the Windows Sockets API (Winsock) that makes the software connection that plugs Windows applications into the TCP/IP stack. As discussed further in Chapter 4, TCP/IP combines the functions of various OSI layers to result in four layers.

Layer:
User
Application
OS
BIOS
Hardware

Figure 3.1 System layers for a standalone PC.

Layer	Function
7 Application layer	What users see
6 Presentation layer	Translation formats
5 Session layer	Opening and closing of sessions
4 Transport layer	Delivery of packets
3 Network layer	Packets from data layer get addressed and routed
2 Data link layer	Packet formation and error correction
1 Physical layer	Hardware and movement of bits

Figure 3.2 OSI reference model layers.

KEY CONCEPTS AND TERMINOLOGY

There are a few key concepts and terms to introduce before discussing specific network hardware, topologies, and protocols. Network architectures generally are based either on *client-server* or *peer-to-peer* approaches. As the names imply, the roles of clients and servers differ from each other—on one side of the interaction are clients that form requests and on the other, servers that respond to them. Peer networks support connections between one or more computers where each is an equal in maintaining the data-sharing connection, such as a direct connection between two PCs. Each device connected to the network is a *node*. This is very often a *computer*, the first of the three Cs. However, nodes can be computers, printers, hubs, or many other network devices. Networks are divided into *segments*, which are subsets of the nodes managed as a group within the larger network. Segments are bounded by devices such as routers, bridges, or hubs, which control the flow of packets (addressable chunks of digital data) to a group of nodes. Segments optimize network performance by grouping together users and devices that share data.

Nodes are connected through some medium, and the capacity of the medium to carry a signal is its *bandwidth*. In earlier network applications, bandwidth was a measure of the range of frequencies possible with analog electronic communication. In the context of digital technologies, bandwidth is measured in terms of the number of bits transmitted per second. This is analogous to the idea of water moving in a pipe. The measure of bandwidth can be thought of as the “diameter of the pipe”; the larger the diameter, the more volume it can carry per unit of time. Similarly, higher network bandwidth carries higher volumes of data per unit of time. Interdevice communication takes place according to a set of rules referred to as a *protocol*, enforced via software. Internet communication takes place through a channel known as a *port*, which is assigned to the specific client–server application within TCP/IP. Logical ports made through these client–server software assignments are thus virtual in nature and not the same as physical computer communication ports such as the familiar serial, USB, or parallel port connections.

The connections created among nodes may use physical wires, wireless radio signals, or a combination of the two. Wired connections may use shielded coaxial cable (thin wire) or unshielded twisted pair wiring such as that used

for telephone wiring. Wired networks utilize some *topology*, which refers to the physical layout of the wired connections.

Finally, computer networks generally move data across a medium by creating *packets*. Packet switching is a process in which data are broken up into smaller pieces, addressed according to a set of rules, and transmitted across the network. Various network protocols specify the packet type used to carry data; when more than one protocol is in use, the data carried may therefore be another type of network packet; this is referred to as packet *encapsulation*.

NETWORK HARDWARE

There are many dedicated types of hardware involved in the connection between a PC and the Internet. These components include the network interface device on a local computer, which allows it to be connected to the network, and the hubs, switches, bridges, and routers that manage the various network segments and direct network traffic.

The Network Interface

The computers or other devices added to a network need dedicated hardware to support the connection. Connection hardware is usually present in the form of an integrated Ethernet port or is added later with the addition of a separate Network Interface Card (NIC). As with all PC devices, an associated piece of software called a *driver* is required to mediate between the NIC and the OS. In the PC system layers, the applications in use communicate with this driver through an *Application Programming Interface* (API), which allows an application to “talk” to the network.

Ethernet network or cable modem connections are common, and most PCs have an integrated port to support this type of cable connection. All NICs or built-in ports have a hardware address associated with them that is used by the network to identify a node. This is its *Media Access Control* (MAC) address; for an Ethernet port, the MAC address is specifically an *Ethernet address* (EA). This address may be used directly by the network to address packets, or mapped to some other type of address, such as an Internet Protocol (IP) address. The correspondence of the MAC addresses and IP addresses is explained in Chapter 4.

Packet Forwarding and Routing Hardware

How do packets coming to or from some networked PC make their journey to a network destination? For an Ethernet connected PC, the lowest level device the packets might encounter is a *hub*. A hub is an OSI physical layer device enabling multiple connections to a virtual bus or star, which are topologies discussed in the next section. A hub only sees bit streams, not packet addresses; it connects all the devices on a particular network segment and forwards all incoming traffic to all those devices. With a hub, only one device can use the network at a time. If the packets need to move across different network segments or to another LAN using the same protocol, they might use another

lower-level device called a *bridge*. A bridge is a device connecting two different network segments at the OSI data-link level. Bridges can filter packets based on their addresses. The packets might also encounter a “smarter than your average hub” device called a *switch*. A switch on an Ethernet network can actually use the MAC addresses associated with the frames (Ethernet packets) and filter the packet stream; it forwards the packets only to the device with the correct address. This is different from a hub, where all devices get the packets whether they are the intended destination or not; instead the filtering takes place at the destination device. Not only are switches more efficient, but they also allow for better security management. For instance, if a computer on a hub is infected with a worm program and causes network problems, the network administrator may elect to cut off access to the entire hub until the problem is resolved, taking all the computers on the hub offline along with the problem machine. With a switch, control at the individual port connection level is possible, allowing the isolation of the specific MAC address associated with the individual problem host.

Finally, the packets may encounter a *router*, a device designed to forward packets between networks or segments. Routers use IP addresses, and it is their job to determine if the packets are destined for a location within the network or for some outside Internet destination. They also determine the best path for packets to get where they are going. Routers make the first determination by the application of a subnet mask. Once the packet destination is known, the best path to the destination is determined. Routers have software allowing them to “talk” to each other; they broadcast messages and use the responses to determine which networks they are connected to. This information is used to build forwarding tables, allowing the efficient forwarding of packets. How routers apply a subnet mask and use the Address Resolution Protocol (ARP) to manage packet flow is discussed in Chapter 4.

Wireless routers combine many of the functions of a router and a switch with that of a wireless access point. Such devices allow users to create local networks that can share devices such as printers and hard drives or share Internet access among multiple devices. Typical wireless routers have a port to connect to a cable or DSL modem and four Ethernet ports to connect other computers or devices. Many also have one or more USB ports to attach a shared network drive or printer, but a printer connected to the router via a USB port usually requires that the router can also act as a print server. Wireless access performance is dependent on the version of the IEEE 802.11x networking standard employed; these are described later in this chapter. Wireless routers can be single or dual band, operating on one or both of the 2.4 or 5 GHz bands, each of which has tradeoffs regarding speed and range. Router speed ratings can be misleading; for example, a dual band router claiming 900 Mbps assumes a theoretical 450 Mbps on each channel, but actual bandwidth would be less.

WIRED NETWORK TOPOLOGIES

The *connections* needed for a network comprise the second “C” of the three Cs. They are typically a physical cable or wire. Although wireless is becoming a popular solution, wired networks are still a common connection strategy for many businesses, schools, and libraries. Wired networks utilize a *topology* or

a map of how the connections are made, much as a topological map of a subway system shows the various lines and stops. The standard wiring topologies usually utilize a bus, ring, or star scheme but can utilize more complex hybrid approaches such as a *tree* combining a bus and a star, or a partial or full *mesh*, which provides many redundant data paths. Charles (1997) and Molyneux (2003) are good sources for more detailed discussions of LAN topologies.

The simplest network to create is a *peer-to-peer* (P2P) connection between two computers. In this approach, there is no central server and little needed hardware. For example, connecting two computer Ethernet ports with a crossover cable, which looks identical to an Ethernet cable but has a crossed wire arrangement in the connectors at each end, forms a simple P2P network. The PC OS itself handles the connection and facilitates sharing of resources on each host.

In a *bus topology*, each node comes off the connecting backbone wire in a linear series as shown in Figure 3.3. Many simple LANs used a bus topology with thin wire coaxial cable looping through each node to the next. In this topology, all nodes see all traffic.

Ring networks usually utilize token passing; hence, they are known as *token ring* networks. In the ring approach, computers are connected together in a circular arrangement, and a special packet, called the *token*, moves around to each node and determines what device can send data, as shown in Figure 3.4.



Figure 3.3 A bus topology.

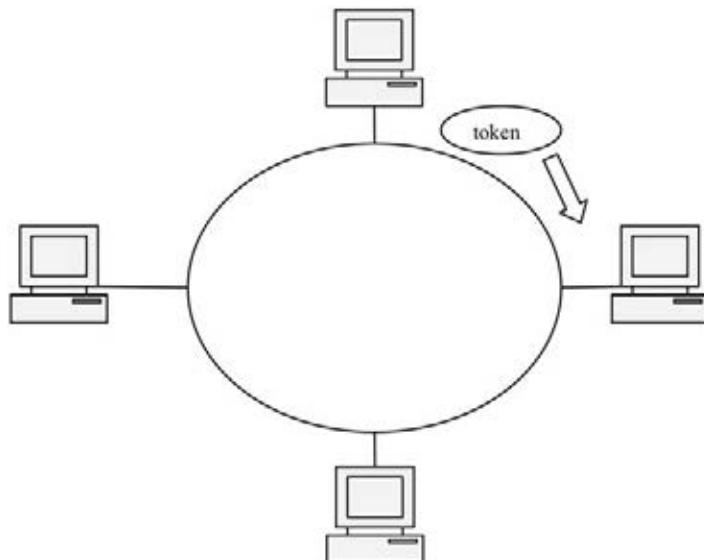


Figure 3.4 A token ring topology.

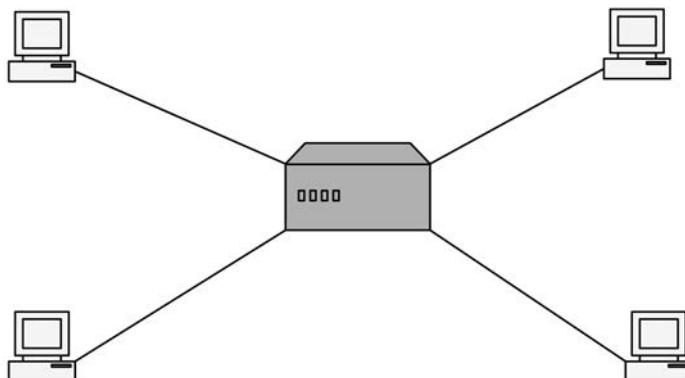


Figure 3.5 A star network topology.

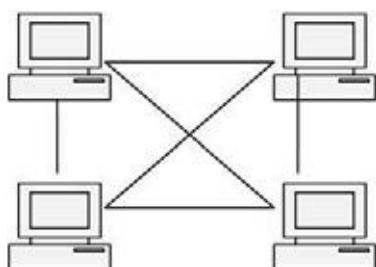


Figure 3.6 A full mesh network.

In a *star* configuration, wires to nodes radiate out from a central server, as shown in Figure 3.5. Many Ethernet networks use a modified star scheme to connect nodes to a central hub.

A *tree* network is a hybrid of a bus and a star, with potentially multiple stars coming off a central bus. It is also possible to combine stars and rings as a hybrid approach. Computer networks can also use a full or partial mesh. In a full mesh, every node is connected to every other node. This ensures direct communication between every pair of devices, but the wiring requirements become quite cumbersome as the number of nodes increases. Figure 3.6 shows an example of a full mesh.

Communication between every pair of devices, but the wiring requirements become quite cumbersome as the number of nodes increases. Figure 3.6 shows an example of a full mesh.

Wired networks utilize twisted pair copper wire or various types of coaxial cable; fiber optic strands are a high-speed connection option. The complex standards establishing the types and gauges of wiring, the distance they can cover and still maintain a reliable connection, conduit specifications, and other requirements must be thoroughly investigated in the network planning stage. However, discussion of the detailed specifications associated with the various types of wired networks is beyond the scope of this overview. Establishing a wired network in an existing older building can have significant hidden costs associated with adding the dedicated conduit needed for some networks. Some network cable is shielded, Ethernet unshielded twisted pair (UTP) wire is not. Unshielded wire is inexpensive and easier to pull through long runs, but it is susceptible to electromagnetic field (EMF) effects that can interfere with data transmission. Moving current in electrical wiring can have inductive effects on the Ethernet wire when they are in close proximity to each other. Therefore, Ethernet UTP wiring is physically isolated from electrical wiring in the runs from the hub to the node locations. Eliminating the need to add conduit is one advantage of wireless networks planned for buildings without Ethernet infrastructure in place, or where it would be prohibitively expensive to install.

WIRELESS NETWORKS

In the previous discussion, the medium carrying the signals was either a thin (coaxial) or twister pair wire, but it is possible, and often desirable, to send packets as signals moving through the air instead. There are well-established successful communication networks predating digital technologies, such as radio and broadcast television but newer wireless technologies include Bluetooth, Wi-Fi, and the various cellular services. Wireless data can be carried by different electromagnetic wave forms on different frequencies, including radio waves, microwaves, or even infrared light waves as with many remote controls. Aside from the obvious advantage of eliminating the need for wiring, Wireless LANs (WLANs) have a number of other distinct advantages. They are low cost, relatively easy to set up, flexible to implement, and quite scalable. For these reasons, they are a good choice for implementations ranging from the small home network to a large business enterprise network. Wireless fidelity, or Wi-Fi, is based on one of the 802.11 options described below and permits connections over a distance of 150–400 feet with speeds of 5–22 Mbps (CNET Wireless Resource Center, 2013a).

As described by Bhola (2002), the roots of today's WLANs go back to a 1971-era network called AlohaNet for data connections among three Hawaiian Islands. Developed by Norman Abramson, it was the first use of radio waves to carry packet transmissions. The connection speeds were relatively slow by today's standards, but this basic idea evolved into the high-speed WLANs of today. WLAN technology began to appear as a local network solution starting in the mid-1990s using proprietary protocols that achieved connections of 1–2 Mbps. The need for a standard similar to Ethernet was apparent, and the IEEE worked with wireless vendors to develop the 802.11 WLAN standards in 1990, capable of 1.0 Mbps or 2.0 Mbps connection speeds. Rapid changes in the supporting technologies drove many extensions to the original standard, beginning in the 1990s with 802.11b, with speeds of 11 Mbps, and 802.11a, which uses higher transmission frequencies permitting speeds up to 54 Mbps. These were followed by 802.11g in 2003 and 802.11n in 2009. Although 802.11g provides speeds of up to 54 Mbps, typical real-world performance is about 22 Mbps. The newer 802.11n not only can provide much higher speeds, but it can also support MIMO (multiple input/multiple output) that provides multiple data paths that utilize different antennas. This results in reduced interference from other devices such as cordless phones also operating in the 2.4 GHz spectrum. The next generation of Wi-Fi is 802.11ac and it offers speeds up to a gigabit per second (Lynn, 2012). This fifth generation of Wi-Fi not only has the potential of speeds that are four times faster than the current standard but also operates on the more open 5 GHz spectrum rather than the increasingly congested 2.4 GHz band (Dave, 2013). A good summary of Wi-Fi generations and technologies is found at the Wi-Fi.org site (Discover and learn, 2013).

Other wireless connections for mobile devices include Bluetooth and cellular services. Bluetooth is a wireless technology with relatively low throughput (about 500 Kbps) and range (about 30 feet) that is used primarily for data exchange between devices such as a keyboard and tablet or a headset and mobile phone.

Cellular services include the third and fourth generation (wireless 3G and 4G) technologies that allow the integration of voice, data, and multimedia applications (CNET Wireless Resource Center, 2013b). Options for mobile data connectivity to the Internet with cellular data services are discussed later in this chapter.

Security is one of the concerns about WLAN connectivity. The idea that your packets are floating around in space for others to intercept certainly is a cause for concern and demands attention. The wireless device *Service Set Identifier* (SSID) can provide a minimum level of access control to a WLAN but is not highly secure because it is not encrypted and is often automatically assigned. This is a useful feature where unsecured access is needed in public spaces such as coffee shops and airports, but ease of connection comes at the expense of security. In addition, such public connections pose the risk of connecting to rogue hotspots that are setup by hackers attempting to lure users in and then stealing passwords or data; these issues are discussed further in Chapter 6. The 802.11 standards include encryption options and these are an important feature; there are horror stories of serious lapses by organizations that have not done enough to secure their wireless networks. For most organizations, especially government agencies or utility companies, ensuring high levels of network security must be a top priority because cyber-attacks are increasing.

Just as Ethernet connections do not necessarily mean an Internet connection exists, WLAN technologies also do not necessarily equate to Internet access. Many devices that have wireless capability cannot provide Internet access outside an Internet connected Wi-Fi hotspot unless they are also 3G capable as discussed later in this chapter.

PROTOCOLS: RULES FOR A COMMON LANGUAGE

The third “C” of the three Cs stands for a *common language*. Many different types of computers are networked together, and each uses its own OS language. How can all these different kinds of computers communicate with each other? They do so by way of network protocols that provides the needed common language. Protocol rules are implemented through software that might be part of the OSs, such as TCP/IP with Windows, or installed later when the network is created. In addition, there may be protocols within protocols, and they may apply to different layers in the OSI model. For instance, the simple network management protocol (SNMP) is part of the TCP/IP suite designed to serve the many types of data, devices, and connectivity at the application level in various network implementations (Cisco Systems, 2008).

There are two fundamentally different approaches to how communication networks can connect nodes; one is *circuit switching* and the other is *packet switching*. These approaches are not mutually exclusive in modern telephone or data networks; for instance, Telco services use a digital switching backbone with analog local loop connections.

Circuit switching has a long history going back to the early telecommunication networks of the nineteenth century. The telegraph and the standard analog telephone service over copper wire are good examples of a circuit-switching

52 Internet Technologies and Information Services

network. You can visualize the idea of this technology by thinking of the old movies with telephone operators working a switchboard, where the operator made a direct circuit connection between callers. This part of the network was then unavailable to other users until they disconnected to free the connection. In contrast, computer networks generally use packet switching, which is a fundamentally different approach. The data are broken up into small, discrete packets and put into a digital “envelope” with a source and a destination address. The packets are transmitted over a network along with lots of other unrelated packets all headed toward their destination. A good physical analogy for this process is the postal service. When a letter is sent by “snail mail,” the data are “encapsulated” in an envelope, which has been given a “From” and “To” address along with appropriate postage, all according to a specific set of rules (or protocols). This physical data packet is dropped into a mailbox along with lots of other unrelated packets and is forwarded to its ultimate destination by a routing process. The Internet depends on a particular packet switching technology called TCP/IP, which is the subject of Chapter 4.

In packet switching communications, the network protocols control packet formation, size, header formats, addressing, and error detection. Because different kinds of networks are often part of a communication hierarchy, packets can carry packets within packets. For instance, an Ethernet packet (aka an Ethernet frame) can carry an IP packet as its data, which in turn carries a TCP packet. There can be different types of packet technologies, each dependent on a network OS (NOS). Many networks use a central network server running a NOS as well as client software on each computer node, and there have been a number of NOS software solutions from different companies over the years. For example, before the current dominance of TCP/IP intranets and Microsoft Windows networking, early PC networks were often Novell-based LANs that used a packet technology called IPX/SPX (Internetwork Packet Exchange/ Sequenced Packet Exchange). When networks that use different protocols are joined together, specialized devices called gateways reside at that interface to handle the needed protocol translation.

ETHERNET

Ethernet is a commonplace network standard used in many organizations. Although Ethernet networks often provide access to the Internet, they are not synonymous; Ethernet can be the medium by which an Internet connection is made, but Ethernet access alone does not provide Internet access. The Ethernet standard and TCP/IP interact with very distinct layers in the OSI reference model; Ethernet functions at the lowest OSI levels, namely the physical and data link layers. The name itself comes from this physical communication medium function; the *ether* of *Ethernet* is a reference to the pervasive ether nineteenth-century physicists erroneously postulated as a necessary medium for light waves to travel through otherwise empty space.

What follows is a brief review of Ethernet networks; Bhola (2002), Simoneau (1997), and Molyneux (2003) are good sources for additional detail on both the history and use of Ethernet networks. Robert Metcalfe developed the first Ethernet specification at Xerox Palo Alto Research Center (PARC) in 1973. This was revised in 1982 and released as Ethernet II by Digital Equipment Corp,

Intel, and Xerox (this is also referred to as DIX Ethernet from the names of the companies involved). The following year, the Institute of Electrical and Electronics Engineers (IEEE) released the 802.3 standard for CSMA/CD networks; CSMA/CD is an Ethernet technology discussed later in this chapter. The number in the name refers to a working group charged in February 1980 to develop this standard; thus “802” is derived from the year and month the charge was given. The later IEEE 802.3 standard is different from the earlier Ethernet II in the structure of the address header. Both headers use 6 bytes (48 bits) for source and destination addresses, but the final two bytes of an Ethernet II header contain “Ether Type” codes for protocol type information, such as one indicating IPv4, whereas the IEEE 802.3 header has packet data length information in its header location. The 802.3 standard supports connection speeds of 1.0 Mbps, 100 Mbps, and 1,000 Mbps.

Wiring specifications are included in the Ethernet standard, and an understanding of the different wiring options is necessary because not all wiring supports all Ethernet speeds. Early versions of Ethernet used both thick and thin wire shielded coaxial cable, but most now use inexpensive and easy to work with UTP (*Unshielded Twisted Pair*) copper wiring similar to that used for telephone wiring, but with different connector jacks. Common Ethernet implementations include 10base-T and 100base-T. The number followed by the word “base” refers to the speed possible and use of a baseband signal; the “T” is for twisted pair. For instance, 10base-T is 10 Mbps of bandwidth with a baseband signal running on UTP wire. This standard has continued to develop, yielding ever-higher speeds with both 100 Mbps (100Base-T) and gigabit Ethernet (1000Base-T) now available. These standards each require different wiring; category 3 (Cat-3) wiring is sufficient for 10Base-T, but category-5 (Cat-5) is required for 100Base-T in order to obtain the maximum bandwidth. This difference became evident on my university network when a hub was upgraded to a newer switch. There was some older Cat-3 wiring in place, which could not support the higher bandwidth now available. The result was the connection speed to those ports had to be reduced at the switch to a level supported by Cat-3. There are also restrictions on the distance Ethernet cables can run; the cable length is usually limited to 100 m unless other devices repeat or boost the signal. In addition to UTP, there is a fiber cable standard for Ethernet.

Besides wiring standards, Ethernet has its own protocols to control packet formation and addressing, as well as technologies to deal with corrupted packets or packet collisions. Because packets share the same connections, collisions are possible, and computer networks need protocols to either prevent, or respond to them. Token ring networks prevent collisions with a special packet called a token that moves through the network and allows only the node with the token to send packets. This is an efficient way to prevent collisions but not a very efficient overall strategy for large networks because any computer with data to send must wait for the token to pass through many nodes with no data to send. Ethernet uses a different approach called CSMA/CD, which stands for carrier sense, multiple access, with collision detection. The carrier signal tells all connected devices they can send data (packets); “multiple access” means many computers can send data at the same time. The protocol detects and resends packets if two computers send packets at the exact same time and they collide. Ethernet packets are frames, and like all packet schemes, there is a fair amount of overhead along with the bit stream of data. Figure 3.7 shows an Ethernet frame.

Preamble (2 bytes)	Destination (6 bytes)	Source (6 bytes)	Type (2 bytes)	Data (46–1,500 bytes)	FCS (4 bytes)
-----------------------	--------------------------	---------------------	-------------------	--------------------------	------------------

Figure 3.7 An Ethernet frame.

As Figure 3.7 shows, an essential part of the Ethernet frame is the six bytes set aside for source and destination addresses. These hardware addresses are called MAC addresses (or Ethernet addresses); every Ethernet frame must record both the source and destination MAC address in the locations shown in Figure 3.7. Six bytes are reserved to represent the MAC address, which means there are 48 bits used for these addresses. From this (and an understanding of binary numbers), it is deduced there are two to the 48th power (2^{48}) possible Ethernet addresses, an astronomically high number. Because people often actually have to work with these addresses for a number of network administration reasons, it is obviously not convenient to deal with 48-bit numbers in binary representation (although computers do not mind at all). To make these long numbers easier to handle, they are expressed in hexadecimal (base 16). This is because binary (base 2) converts easily to hex, and it also makes long binary numbers much more compact. For instance, four binary digits (called a “nybble” or half a byte) could be represented with two decimal digits or a single hexadecimal digit. The equivalency is $1111_{\text{(base 2)}} = 15_{\text{(base 10)}} = f_{\text{(base 16)}}$. Extending this idea, 16 binary digits can be written with four hexadecimal digits, which means the 48-bit Ethernet addresses can be written with three sets of four hexadecimal digits, such as in the hex number: **310b.11e9.f432**.

Because many Ethernet-connected computers are a node on the Internet, the data the Ethernet frame carries might be an IP packet. A protocol called ARP manages correspondence between these two types of addresses, as discussed in Chapter 4.

VIRTUAL PRIVATE NETWORKS

A network can be extended through a *Virtual Private Network* (VPN). This strategy requires a VPN client program to connect to an internal organizational network through a firewall over a public network (usually the Internet). The VPN client and firewall create a virtual, secure “tunnel” through which authentication data and private network traffic can pass, allowing a remote user to access their organization’s network from the outside. Other supporting protocols and standards, such as the Point-to-Point Tunneling Protocol (PPTP), provide differing levels of security and facilitate secure connections.

PROXY SERVERS

Proxy servers can also make the connection between a LAN or a specific host and the Internet. Proxy servers can serve as a firewall to prevent the outside world from directly interacting with any host on the LAN. By using the *Network Address Translation* (NAT) protocol, proxy servers can expand an

IP address class by managing the correspondence between private IP address space and a single public IP for packets destined to the outside world. Proxy servers are also used to present an authorized IP address to some service that requires authentication of users accessing restricted resources. Libraries frequently manage access to certain resources by defining an authorized IP address space in a domain. Users accessing the services from within this defined address space, such as on-campus users of an academic library, automatically have access. Off-site users must first authenticate themselves to a proxy server that then presents the appropriate IP address when the request is forwarded to the service. Libraries often manage this process with OCLC's EZproxy software ("EZproxy authentication and access software," 2014). These devices and strategies are discussed in more detail in Chapter 4.

CONNECTING TO THE INTERNET

Internet access requires that a connection technology be available and properly configured. Historically, access to such connection infrastructure was a major limiting factor to Internet participation. In the days when the "information superhighway" metaphor was in use, people talked about the problem of the "last mile" to reflect that the Internet was not available in many areas for every home, school, and library. Although we no longer often hear the phrase "information superhighway" now, the idea of the "last mile" remains, because it still presents a significant barrier for many. Even though broadband access has increased, it is not always available, particularly in rural areas where the only options might be either older dial-up connections or satellite or cell service (Gilroy & Kruger, 2006). The bandwidth intensive activities of today require broadband services, and the lack of universal, affordable broadband makes the digital divide a very real problem for many.

During much of the early history of the Internet, direct access was not commonly available, and users often connected through an intermediary shell account on a mainframe or UNIX system. Fortunately, some form of locally available Internet access has become ubiquitous in most areas, with many ISPs competing to provide Internet connectivity. Historically, common options for connecting to the Internet typically have involved one of the following strategies: a POTS dial-up connection, a leased dedicated telephone link such as a T1 or fractional T1 line; digital telephone services such as Integrated Services Digital Network (ISDN) or Digital Subscriber Line (DSL); broadband cable service connections; or a cellular service data plan. Although dial-up connections still exist, they have mostly disappeared except in rural or low income markets. The proliferation of both the connecting infrastructures and ISPs has made it much easier to make a direct Internet connection through dial up, DSL, or cable. This section focuses on these connection technology options; how they result in the assignment of an Internet address will be covered in Chapter 4.

The Internet Service Provider (ISP)

The ISP's job is to provide an IP address. The how and why of that process is discussed in detail in the next chapter. ISPs manage and assign IP addresses

to their customers, acting as their gateway to the Internet. However, who provides this service to the providers? The early NSFNET contracted with commercial telecommunication companies to service the four Network Access Points (NAPs) where Network Service Providers would exchange Internet traffic. Today we have IXPs where ISPs exchange traffic among networks. There are a small number of Tier 1 providers, defined as a network that can reach every other network without purchasing IP transit or paying settlements (Who provides Internet service, 2013). These include large, multinational corporations such as AT&T, Centurylink, Verizon Business, Sprint, Level 3 Communications, Inteliquent, Deutsche Telekom AG from Germany, TeliaSonera International Carrier from Sweden, and NTT Communications from Japan. Most Internet traffic requires at least several “hops” and Tier 1 networks have voluntary arrangements to carry each other’s traffic in a peering arrangement. Tier 2 networks have some peering arrangements but also act as resellers of Internet access; Tier 3 networks only purchase IP transit, usually from Tier 2 networks, and resell access to us as an ISP.

Modems

As introduced in Chapter 1, a modem (modulator/demodulator) is a device that allows a computer to connect to other computers; initially this was via an analog telephone line but modems are also used now to make computer connections via cable services as well. These devices were in common use even before there was interest in connecting to the Internet; they were used to facilitate access to mainframe or UNIX hosts, to connect to online database providers, or to connect to computer bulletin board services. Modems convert analog signals to digital and digital to analog. The modem UART (Universal Asynchronous Receiver/Transmitter) chip converts the parallel data stream on a computer bus into a serial data stream for communication as shown in Figure 3.8. Modem-to-modem communication requires communication parameters, which are the number of data bits, stop bits, and parity bits used to transmit the data. Phone line modems exchange these parameters during the initial “handshake” resulting in the “squawking” tones heard at the beginning of a dial-up session. Early modem speeds were expressed as the baud rate, which is not exactly the same as bps (bits per second). Baud rate is a measure of switching speed, equivalent to the number of frequency changes per second the modem can perform, which approximates bps. Modern modems measure bandwidth in the more familiar bps measure. The 56 Kbps modem became standard for dial-up connections, but this 56 K speed is what the device is theoretically capable of, not the actual speed of an analog data connection. The Federal Communications Commission (FCC) limits the maximum permitted speed for this connection to not more than 53 K to prevent “cross talk” (where a signal can “bleed” across a copper line and cause noise and interference). For this and other reasons, 56 K modems typically resulted in connection speeds in the 28.8–45 K range (56 K info, 2007; Gilroy & Kruger, 2006). This is far less bandwidth than is possible with either DSL or cable connections, and the use of dial-up connections for Internet access among American adults has declined from a high of 41 percent in 2001 to about 4 percent in 2012 (Breener & Lee, 2012).

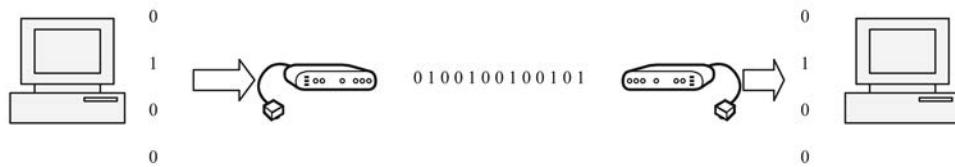


Figure 3.8 The modem's UART chip converts the computer parallel bit path into a serial one for transmission. The communication parameters would typically include additional overhead such as the insertion of start and stop bits surrounding each byte. At the receiving end, the UART converts the serial bit stream back to a parallel one.

Analog Phone Service: POTS and PSTN

Modems were designed to work with POTS (Plain Old Telephone Service). This refers to analog signals moving on copper twisted pair wire over the Public Switched Telephone Network (PSTN). POTS was designed to carry an acoustic signal by converting it to an electrical signal with volume (signal amplitude) and pitch (signal frequency). Telephone networks (at the local level at least) are an example of circuit switching technology; this is the Public Switched Telephone Network (PSTN). They do use digital technology for the backbone service and trunk lines, but the connection from most homes to the local telephone company central office, referred to as the local loop, is usually analog. Although an analog connection works fine for voice communication, it is decidedly limited for transmitting digital data. Several technologies, collectively known as broadband, better accommodate both types of communication with much higher connection speeds. Broadband options include cable services, ISDN, and the many forms of DSL.

Broadband Services

Broadband is a generic term used for the various high bandwidth connection technologies. By the end of 2012, about 65 percent of all U.S. online households were using a form of broadband at home compared to just 4 percent in 2001 (Brenner & Lee, 2012). Cable modem (CM) access has had an early lead over DSL; in 2004, cable had 75 percent of the broadband market compared to 15 percent for DSL. For a time, DSL was expected to make gains in this area over time based partly on the extensive infrastructure that the Telcos had compared to cable companies, especially in rural areas (Wrolstad, 2004). However, it now seems that cable has become the leader in home broadband services as the telcos have been dramatically losing broadband market share while cable companies are gaining (Malik, 2012). The telcos now appear to be focused on developing their cellular service networks rather than with promoting DSL services; in fact, in some rural areas they are seeking regulatory relief to abandon landline service altogether (Higginbotham, 2009). Both DSL and cable connections now compete with the older, but still available, leased phone line options such as T1, fractional T1, or T3 lines.

Leased Telephone Lines

One of the main early options for a high-speed connection to the Internet was to lease a dedicated line. AT&T developed the T1 line in the 1960s, so this technology has been available for some time. T1 lines have 24 64 Kbps channels providing 1.5 Mbps of bandwidth, about the same as a standard DSL connection. The T1 bandwidth can be subdivided and sold as a fractional T1 connection of 128 Kbps or other multiples of 64 Kbps. The T3 lines have a bandwidth of about 45 Mbps or about 30 times the capacity of a T1. The backbone service for the Internet was upgraded to T3 lines in 1992. It is interesting to note that the original NSFNET backbone service for the Internet used T1 lines until 1991, so the Internet backbone connections of that time were about the same speed as the average home DSL connection of today (National Science Foundation, 2005).

Integrated Services Digital Network

Integrated Services Digital Network (ISDN), developed in the mid-1980s, was a digital precursor to DSL. It allowed for faster connections and used multiple channels for connecting multiple devices, such as telephones, faxes, and computers. Data rates of 128 Kbps were possible along with two 64 Kbps “B” channels and a 16 Kbps “D” channel. However, there were often problems implementing ISDN technologies; these included inconsistent local telephone company support, the need for specialized nonanalog devices, and interoperability issues because there was no standardization of the supporting telephone company switching technologies. One advantage of ISDN was that it provided very fast call “set up” and “tear down,” which refers to the “handshake” period when modems make a connection to other modems. The ISDN could accomplish this in milliseconds, allowing for very rapid connects and disconnects to and from the Internet. This was an initial advantage because early ISP cost models often charged by the minute for online time, and it was therefore desirable to minimize connect time. Being able to seamlessly connect, disconnect, and then quickly reconnect with no noticeable delay was possible with ISDN. Although ISDN overcame some of its initial problems, various forms of DSL and cable connections have become the common choice for broadband access.

The per-minute charge model that gave ISDN an edge is not common today. Most service providers provide unlimited home access for a set fee. However, as more users download movies and other large entertainment files, some major service providers are reevaluating this access model. Service providers Comcast and AT&T recently announced that they intend to initiate “Internet metering” that charges heavy-use customers a surcharge when a set bandwidth limit is exceeded. These variable surcharges could create consumer anxiety about the cost of their Internet activities, and Internet developer and Google executive Vint Cerf believes this change would hamper innovation and application development (Stelter, 2008). Now that the FCC has proposed rules to allow ISPs to offer faster speeds and variable pricing for some content it will be interesting to see if such a metered approach becomes the norm for Internet access.

Digital Subscriber Line

Digital Subscriber Line (DSL) technology has become more pervasive; it has most of the advantages of ISDN for lower cost and faster speeds. DSL is one viable solution to the “last-mile” problem alluded to earlier because it provides high-speed, local loop connections over existing copper wire infrastructure (Sheldon, 2001). DSL dedicates part of the available bandwidth to support an analog voice signal, which allows simultaneous computer and voice connection. Initially, DSL appealed primarily to small businesses as an attractive and affordable Internet access alternative to the more expensive T1 or fractional T1 line option. As prices have declined, DSL connections have become competitive with cable TV broadband connections for individual home use.

DSL is a generic term for a large variety of specific standards that fall into one of two basic types: symmetric, where the upstream and downstream speeds are the same, and asymmetric, where they are different. There are a huge number of different specifications for DSL including DSL Lite, High bit-rate DSL, Symmetric DSL, Asymmetric DSL, and Very High DSL (BytePile, 2002; Tyson, 2013).

Because DSL is a dedicated point-to-point connection, no dial-up has to take place. This results in an ongoing Internet connection; this is convenient, but it has possible security implications for the user. Many home connections make use of some form of Asymmetric Digital Subscriber Line (ADSL). As noted previously, *asymmetric* refers to the difference in bandwidth between the “upstream” and “downstream” connection. Most users of the Internet care more about the speed of the connection coming to the computer than the connection going away. This makes sense in the context of HTTP, where the request header generated by the client software is quite small but results in the delivery of a number of large files to the client. With an asymmetric connection, you might have from 2.5 to 10 Mbps of bandwidth to your computer, but only 768 Kbps from it. There are many levels of bandwidth combinations available depending on the service provider and your budget. Availability of DSL is in part constrained by the distance to a phone switching facility; there is a maximum 3.4-mile (5.5 km) distance limit, and beyond that distance there is the added cost of a repeater to boost the signal.

Cable Broadband

Television cable systems have become important providers of broadband Internet connections. The cable infrastructure is in place in most cities, and the number of people using cable services is high. Cable broadband connections require a specialized interface device (a cable modem) and a computer or router with an Ethernet port for the connection. Internet access through cable service has its own set of tradeoffs. The cost is generally competitive with DSL, but availability can be an issue. Connection speeds are comparable and often better than DSL, but cable was not designed to do two-way traffic, so as with ADSL, the bandwidth is also asymmetric with downstream rates often of 10–20 Mbps vs. upstream rates of about 300 Kbps. As with DSL, this is not a significant issue for most Internet users unless they are running a server or other service that may require more upstream bandwidth. Another issue for cable

modem users is that the nodes in an area or neighborhood share the available bandwidth, so connection speeds can degrade with additional subscribers. Like DSL, cable connections result in a computer constantly connected to the Internet, making Internet use convenient but raising security issues as well.

Both DSL and cable technologies provide affordable and fast connections to the Internet, and for some time they were each gaining wide acceptance (Wrolstad, 2004). However, although DSL initially competed successfully with cable, it appears that cable has won the Internet access race (Crawford, 2013). In a single quarter of 2011, Verizon and AT&T lost over 100,000 and 600,000 DSL line subscribers, respectively, while Time-Warner Cable gained 130,000 connections (Malik, 2012). Where both are still options, the decision about which to choose often comes down to local availability, pricing, and other package options (Jupiter Media, 2005).

Satellite Internet Service

An alternative to dial up or other connections that utilize landlines is satellite Internet service. These services depend on geostationary satellites and a receiving dish just as with satellite television, but require a satellite modem and sometimes the installation of a second separate satellite dish. Satellite Internet access is relatively slow with maximum downstream speeds of 1–1.5 Mbps and upstream speeds in the 100–256 Kbps range, but the actual speeds can be much slower and are subject to latency delays and possible interference due to weather (Parsons & Oja, 2013). However, they are sometimes the only viable option in rural areas.

Fiber Options: FiOS Internet, U-Verse, and Google

In addition to DSL options, both AT&T and Verizon have promoted fiber broadband solutions; Verizon, along with Frontier Communication, offer FiOS (Fiber Optic Service); AT&T's option is called U-Verse. These are point-to-multipoint passive optical networks that use optical splitters to distribute a signal to multiple homes or neighborhoods with a single fiber-to-the-premises or fiber-to-the-node connection. The bandwidth can be distributed from a node or within a home with existing wiring for phone, video, and Internet access. Google entered the fiber marketplace in 2009 with the announcement of a competition to select trial cities in which they would begin building a Gigabit fiber network. Over 1,100 cities applied and Kansas City, Kansas, was selected in 2011 (Adams, 2013). Google has plans for continuing in another dozen cities (Welcome to Google cities, 2013). Other fiber projects are supported by federal government broadband stimulus grants to local governments. One such project is in Cook County Minnesota in the rugged northeastern region of the state where the Arrowhead Electric Cooperative is laying cable to connect all residents (Peters, 2011).

Fiber options provide speeds of 100 Mbps to 1 Gbps but the availability of these services is limited and building fiber infrastructure is expensive—as of this writing, Google has reportedly spent nearly 100 million dollars to launch the Kansas City fiber network (Graziano, 2013). In addition, some believe a tacit agreement has emerged among the major telcos and cable companies not

to directly compete with each other in the same market (Crawford, 2013). It also appears that telcos are ceding wired customers to cable providers to focus instead on offering wireless services such as 4G LTE (Higginbotham, 2011). Consequently, high-speed fiber network services have not achieved the market penetration of either DSL or cable. Although some question whether such high capacity bandwidth is really necessary or worth the high capital investment costs, others counter that investing in high-speed fiber networks is critically important, not just for entertainment, but for advanced medical services, education, and economic development. The United States is lagging other countries in fiber network deployment; fewer than 8 percent of Americans have fiber service compared to much higher levels in countries such as South Korea, Japan, and Australia (Crawford, 2013).

Another technology that is of interest is BPL (*Broadband over Power Lines*) that uses the existing AC power infrastructure in combination with fiber to deliver data services to homes and businesses along with electric power. The data signal bypasses the high-voltage parts of the grid using fiber or wireless links to deliver the data signal to the medium voltage lines that can reliably support them. Once delivered to the power pole transformer, data services can then be distributed over the existing lower voltage wiring to the structure or via a wireless link from a transmitter at the power pole (Mollenkopf, 2004). If the power wire is used to deliver the data service, a BPL modem plugs into a standard AC outlet and provides a standard Ethernet port as with other types of broadband modems. BPL faces a number of technical and regulatory challenges regarding FCC licensing of the necessary devices and potential signal interference and degradation, but given the well-established power grid, it is an interesting possibility.

Wi-Fi Broadband

A Pew Research Center survey reports that by 2010 more than 59 percent of American adults used a wireless computer or cell phone for Internet access (Smith, 2010). Organizations and individuals interested in accessing the Internet within a local area are increasingly turning to wireless technologies. One option for wireless broadband Internet access is Wi-Fi, which stands for *wireless fidelity*, an IEEE standard for wireless connectivity by way of a broadcast signal using Wireless Application Protocol (WAP). The various versions of this technology are derived from the IEEE 802.11 standard (other variants of this standard are covered earlier in this chapter). Wireless technologies are very viable solutions for areas where wiring is problematic or undesirable, such as older buildings, homes, and coffee shops. Wi-Fi requires an access device or router connected to a broadband source, which in turn provides access to any device with a wireless network adapter. Security has been a concern with wireless Internet access because it is possible to intercept these signals or break into the network. Wireless security was initially addressed with Wired Equivalent Privacy (WEP), an early standard that was displaced by Wi-Fi Protected Access (WPA) and now with WPA2 for higher levels of security.

Wi-Fi is not the same as cell service; this can be confusing because some mobile phones can switch over to “free” Wi-Fi when it is available. It is important to recognize that many wireless devices can access the Internet only when

they are within the narrow geographical range of a wireless access point unless it also has cellular data service.

MOBILE AND CELLULAR DATA SERVICES

In 2010, smartphone shipments surpassed those of the long dominant PC (Goldman, 2011). By 2012, nearly half of American adults had smartphones and the numbers continue to grow across all demographic groups; young and old, men and women, rich or poor, urban or rural, we are increasingly dependent on these devices for Internet access (Smith, 2012). The global numbers are even more dramatic, with much of the world being even more dependent on mobile devices than the United States. There are many platforms, service plans, and technologies to choose from, and having a smartphone is of little use without the bandwidth needed for the data-intensive activities of the Web. Although most cellular data access is to enable Internet connections for smartphones, other devices such as tablets can have built-in support for cellular data plans or can be setup for it with the addition of a plugin wireless USB modem. Another option is to use an attached smartphone as a modem. In addition, some phones, as well as devices such as the MiFi wireless router from Novatel Wireless, can be used to establish small wireless network hotspots anywhere there is cell service (Parsons & Oja, 2013).

Mobile Phones

The mobile phone has quickly evolved from being an impractical, expensive novelty to an extremely portable, versatile computing and communication tool. Fling (2009) summarizes the relatively short history of the cell phone and describes it in the context of five distinct eras to date:

1973–1988	The “brick” era with “suitcase” size phones with very large handsets and limited range.
1988–1998	The “candy bar” era with much smaller rectangular handset with short antenna, much improved from the first generation but they are still expensive and larger than today’s. Voice and text services are available, but these were not often used as a primary sole phone.
1998–2008	The “feature phone” era with smaller, often “flip” style phones with added features such as a camera and the first apps.
2002–present	(overlaps with the “touch” era below): The “smartphone” era with a mobile OS, a QWERTY keyboard, and larger screen that makes mobile Web access practical.
2007–present	The “touch” era—Apple introduces the iPhone, which becomes wildly successful. A true phone/computer with many apps and an improved user interface using touch screens and voice commands.

Mobile Layers

The OSI reference model discussed earlier in this chapter described a hierarchy of layers, each of which has a specific function and communicates with adjacent layers. As Fling (2009) explains, our interaction with mobile devices and the services available with them also takes place within mobile layers that depend on each other, shown in Figure 3.9.

Part of the complexity of the mobile environment is because there are different competing players involved with each of these layers. Operators, also referred to as carriers, provide access to their network by installing cell towers and operating a cell network that can deliver services such as Internet access. Devices, most often now a smartphone, communicate with the network through a platform or core programming environment provided by the device maker, which may be the bundled OS itself. Platforms can run services on a device, such as browsing the Web, using GPS and map applications, or sending a text. OSs run applications directly or via an application framework such as Java ME. An application framework refers to the first layer developers can work with, providing an Application Program Interface (API). Mobile platforms and OSs can be licensed, such as Java Micro Edition, proprietary, such as the iPhone OS, or Open Source, such as Android.

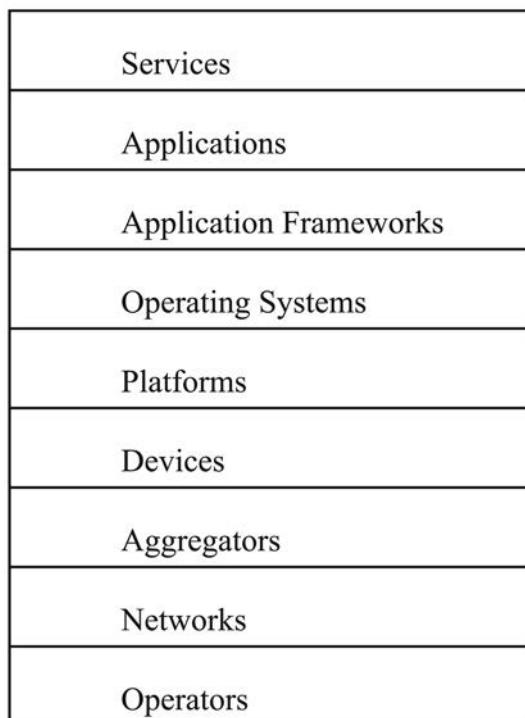


Figure 3.9 Mobile layers (Fling, 2009)

Data Services

Bandwidth and a reliable connection are critical for successful access to the mobile Web and there are a sometimes-confusing array of connection options and service plans. Current advertising for mobile services emphasizes that “faster is better,” which few would argue with, but how fast an individual connection will be can vary considerably. Most of the services offered today refer to 3G or 4G speeds; however, 3G is not a single standard but represents a number of different connection technologies. The “G” refers to the various generations of mobile, summarized in Table 3.1.

The first “generation” for cell phone services goes back to 1973 using analog radio waves; this was retroactively labeled “1st” generation. 2G emerged in 1991 and was digitally encrypted. Data services were possible with 2G, but both these early generations were designed for voice communication. 3G was released in 2001; it is important to recognize that 3G represents many technologies and is not a single standard or speed. The various 3G technologies include Wideband Code Division Multiple Access (W-CDMA), Universal Mobile Telecom System (UMTS), and High-speed Packet Access (HSPA) with associated speeds of 3.6–14.4 MB/s (Fling, 2009).

Improvements to 3G include 4G LTE (for long-term evolution) and WiMax, developed by Sprint Nextel (Wong, 2008). WiMax stands for Worldwide

TABLE 3.1 3G technologies and speeds.

Generation	Date	Foundation
1st “G”	1973	Analog radio waves; retroactively labeled “1G”/ Speeds of 9.6 to 14.4 Kbps
2G	1991	Digitally encrypted; used GSM (Global System for Mobile Communications, based on TDMA (Time division multiple access). First data services, but designed for voice Speeds of 9.6 to 115 Kbps
2.5G	2001	A transition technology with speeds of 56 to 144 Kbps)
3G	2001	A number of technologies and standards of varying speeds based on UMTS (Universal Mobile Telecommunications System). WiMAX (Worldwide Interoperability for Microwave Access) added in 2007 Speeds of 55 to 614 Kbps
4G	2008	IMT-Advanced (International Mobile Telecommunications Advanced) network based on an all IP packet-switched network Speeds of 20 to 40 Mbps

Interoperability for Microwave Access. 4G LTE is one of several 4G wireless communication standards developed by the 3rd Generation Partnership Project (3GPP) that is ten times faster than 3G (4G LTE, 2013). WiMax is the IEEE 802.6 standard that uses a licensed channel on the radio spectrum, as opposed to the free frequencies used for Wi-Fi, and it provides a fivefold increase in connection speed that provides a stronger, cleaner, and more secure signal that is less susceptible to interference. Samsung has recently announced a research breakthrough for mobile with the development of “fifth-generation” networks that use a broad band of frequencies that are expected to be much faster than current 4G speeds; initial tests delivered 512 Mbps compared to the current 75 Mbps speed of 4G LTE (Talbot, 2013). Samsung hopes to commercialize these technologies by 2020 to allow users to transmit huge files such as high-definition movies. Whether this prediction happens or not, it is clear that mobile connection technologies will continue to provide increased capacity.

As much as wireless technologies have improved, at current speeds they cannot compete with other broadband technologies such as cable or fiber networks, which are as much as a hundred times faster. They are also subject to interference, and transmissions at high frequencies can be blocked by buildings or other topographical features of the landscape. For many, wireless is a complementary service that is used in combination with other connection services. Those whose only option is wireless may find it is inadequate for many important educational, medical, or entertainment activities (Crawford, 2013).

BROADBAND TECHNOLOGIES AND THE DIGITAL DIVIDE

Net neutrality was introduced in Chapter 1 and it is an important part of the connectivity discussion. The core idea of net neutrality is that all packets should be given equal priority; whether the datagrams carry an email or a video-streamed movie they are treated the same with none being given priority over another. The other side of this debate suggests that such tiered access is both fair and reasonable; some packets should be given priority and it should be based on the customer’s willingness to pay. In this view, the “information superhighway” should have lanes where some traffic is routed more quickly to some users. The discussion of connection technologies in this chapter is highly relevant to this debate because the regulatory environment has different rules depending on how a technology is defined; for example, there is a distinction between a telecommunication service and an information service. Telecommunications are controlled by the FCC; early telecommunication law defined telcos as common carriers that were obligated to treat everyone equally, in part to ensure that all Americans had equal access to telephone service. Many telcos are now seeking to free themselves from the obligation to provide universal landline services, which would force those customers to either sign on to more expensive mobile plans or do without. In my state of Kentucky, AT&T is advocating for legislation that would allow the discontinuation of landline service in rural areas where, despite claims to the contrary, mobile services can be unreliable. Wireless carriers seek to have their mobile networks treated as being equivalent to wired broadband. However, although net neutrality rules for all Internet access are changing, earlier restrictions did not apply to wireless. Wireless

carriers could always prohibit, or charge more, for bandwidth intensive activities such as peer-to-peer file sharing or video streaming. Many data plans for mobile access already have limits on monthly data use, but so far, these have not applied to those with wired cable or DSL Internet access at home.

The digital divide refers to the gulf that exists between those who have reasonable and relatively convenient access to digital content and those who do not. There are many socioeconomic and educational facets to this problem, and although there is disagreement on the extent and causes of the problem, everyone agrees that those without Internet access are at a serious disadvantage. Concerns about the potential of a digital divide between those who have access and those who do not have been concurrent with the emergence and growth of the Internet, but the divide is now not just about having access, but is about the kind of access. Broadband is essential for most Web activities and the potential loss of landline and other wired options for rural citizens eliminates important connection options. Libraries can help mitigate these disparities by providing broadband services. Proposed changes to the federal E-rate program could result in increased funding for broadband for schools and libraries and better connection options for the communities they serve. A survey by the American Library Association found that 60 percent of public libraries reported an increase in demand for computers and Wi-Fi access in 2011–2012, but many also reported bandwidth that was inadequate for user needs—only about 30 percent of public libraries reported connection speeds of 10 Mbps or greater (U.S. public libraries, 2012).

SUMMARY

Networking technologies have greatly enhanced and extended the productivity potential of all computing devices, and the PC would not be in almost every home and school if it were not for the resource and information-sharing capabilities networks provide. This capability comes at a cost for specialized hardware and software, local infrastructure, and network administrative personnel. The network fundamentals explored here, such as the OSI reference model and the packet technologies networks depend on, form an essential backdrop to the chapters that follow on Internet technologies based on TCP/IP.

The strategies and technologies supporting Internet connections still vary considerably and depend, in part, on whether you are connecting an individual personal computer or a LAN to the Internet. Early on, direct connections to the Internet were rare, and sessions were often initiated via a shell account on some Internet host such as a University mainframe or UNIX system. With the emergence of TCP/IP-enabled PCs and competing ISPs, direct Internet access became possible, first with analog dial-up connections and now with various broadband technologies. Wireless access is becoming quite commonplace in many public spaces and has become a primary network strategy for many organizations. Mobile devices using 3G and 4G data services continue to displace wired PCs as the primary way many users connect to the Internet.

Internet connection technologies include analog and digital telephone lines, fiber optic networks, cable company broadband, Ethernet, Wi-Fi hotspots, and cell service. Each option has its own set of advantages and disadvantages; the

final choice is determined by availability, the price of the various broadband services, and bandwidth requirements of the activities planned. The decline in wired services in many areas and the increased use of wireless services and smartphones has intensified the net neutrality debate.

REFERENCES

- Adams, Shar. (2013, June 4). The business of Google Fiber. Retrieved June 4, 2013, from <http://www.theepochtimes.com/n3/89237-the-business-of-google-fiber/>
- Bhola, J. (2002). *Wireless LANs demystified*. New York: McGraw-Hill.
- Brenner, Joanna, & Rainie, Lee. (2012, December 9). Pew Internet: Broadband. Retrieved June 3, 2013, from <http://pewinternet.org/Commentary/2012/May/Pew-Internet-Broadband.aspx>.
- BytePile.com. (2002, October 19). DSL categories. Retrieved August 31, 2007, from http://www.bytespile.com/dsl_categories.php.
- Charles, G. T., Jr. (1997). *LAN blueprints*. New York: McGraw-Hill.
- Cisco Systems. (2008). Simple network management protocol (SNMP). In *Internet-working technology handbook*. Retrieved May 1, 2008, from <http://www.cisco.com/en/US/docs/internetworking/technology/handbook/SNMP.html>.
- CNET Wireless Resource Center. (2013a). Wi-Fi. Retrieved July 18, 2013, from http://www.cnet.com/1990-7363_1-6361076-3.html.
- CNET Wireless Resource Center. (2013b). 3G. Retrieved July 18, 2013, from http://www.cnet.com/1990-7363_1-6361076-4.html.
- Crawford, Susan P. (2013). *Captive audience: The telecom industry and monopoly power in the new gilded age*. New Haven, CT: Yale University Press.
- Dave, Paresh. (2013, June 22). Wi-Fi is about to get a lot faster and more reliable. Retrieved July 18, 2013, from <http://articles.latimes.com/2013/jun/22/business/la-fi-tech-savvy-wifi-20130622>.
- Discover and learn. (2013). Retrieved May 30, 2013, from <http://www.wi-fi.org/discover-and-learn>.
- 56 K Info. (2007). Technical support. Retrieved May 1, 2008, from <http://home.core.com/web/technicalsupport/library/56kinfo.html>.
- Fling, Brian. (2009). *Mobile design and development* (1st ed.). Sebastopol, CA: O'Reilly.
- 4G LTE. (2013). Retrieved June 4, 2013, from http://www.webopedia.com/TERM/4/4G_LTE.html.
- EZproxy® authentication and access software. (2014). Retrieved May 9, 2014, from <http://www.oclc.org/ezproxy.en.html>.
- Gilroy, A. A., & Kruger, L. G. (2006). Broadband Internet regulation and access: Background and issues. Retrieved October 1, 2007, from <http://usinfo.state.gov/infousa/economy/technology/docs/60574.pdf>.
- Goldman, David. (2011, February 9). Smartphones have conquered PCs. Retrieved February 10, 2011, from http://money.cnn.com/2011/02/09/technology/smartphones_eclipse_pcs/index.htm?source=cnn_bin&hpt=Sbin.
- Graziano, Dan. (2013, April 8). Google Fiber has cost less than \$100 million to launch so far from <http://bgr.com/2013/04/08/google-fiber-cost-nationwide-423644/>
- Higginbotham, Stacey. (2009, April 17). Carriers aim to keep rural broadband under their thumb. Retrieved June 4, 2013, from <http://gigaom.com/2009/04/17/carriers-aim-to-keep-rural-broadband-under-their-thumb/>
- Higginbotham, Stacey. (2011, July 21). Since DSL is obsolete, AT&T will sell you LTE instead. Retrieved June 4, 2013, from <http://gigaom.com/2011/07/21/since-dsl-is-obsolete-att-will-sell-you-lte-instead/>

- Jupiter Media. (2005). JupiterResearch forecasts broadband's rise to dominance, increasing from 32 million U.S. households in 2004 to 69 million in 2010. Retrieved October, 2007, from <http://www.jupitermedia.com/corporate/releases/05.06.02-newjupresearch.html>.
- Lynn, Samara. (2012, November 8). How to buy a wireless router. Retrieved May 30, 2013, from <http://www.pc当地.com/article2/0,2817,2347539,00.asp>.
- Malik, Om. (2012, January 6). The continued decline of DSL. Retrieved June 3, 2013, from <http://gigaom.com/2012/01/26/the-continued-decline-of-dsl/>
- Mollenkopf, Jim (2004, September 23). Presentation to Cincinnati IEEE Meeting. Retrieved July 18, 2013 from http://ieee.cincinnati.fuse.net/BPL_slide_show.pdf.
- Molyneux, R. E. (2003). *The Internet under the hood: An introduction to network technologies for information professionals*. Westport, CT: Libraries Unlimited.
- National Science Foundation. (2005, September 28). The launch of NSFNET. Retrieved October 1, 2007, from <http://www.nsf.gov/about/history/nsf0050/internet/launch.htm>.
- Parsons, June, & Oja, Dan. (2012). *NP on Computer Concepts 2013 Comprehensive*. Boston, MA: Cengage Course Technology.
- Peters, Dave. (2011, July 12). Broadband 7 update: Fiber hard to find and bids high but construction starts. Retrieved August 15, 2013, from <http://blogs.mprnews.org/ground-level/2011/07/broadband-7-revisited/>
- Sheldon, T. (2001). DSL (digital subscriber line). *Tom Sheldon's Linktionary*. Retrieved January 27, 2006, from <http://www.linktionary.com/d/dsl.html>.
- Simoneau, P. (1997). *Hands-on TCP/IP*. New York: McGraw-Hill.
- Smith, Aaron. (2010, July 7). Mobile access 2010. Retrieved May 30, 2013, from <http://www.pewinternet.org/Reports/2010/Mobile-Access-2010.aspx>.
- Smith, Aaron. (2012, March 1). Nearly half of American adults are smartphone owners. Retrieved May 30, 2013, from <http://www.pewinternet.org/Reports/2012/Smartphone-Update-2012.aspx>.
- Stelter, B. (2008, June 15). Internet providers clamp down on users' time online. *Lexington Herald-Leader*, p. A7.
- Talbot, David. (2013, June 3). Samsung says new superfast "5G" works with handsets in motion. Retrieved June 4, 2013, from <http://www.technologyreview.com/news/515631/samsung-says-new-superfast-5g-works-with-handsets-in-motion/>
- Tang, A., & Scoggins, S. (1992). *Open networking with OSI*. Englewood, NJ: Prentice-Hall.
- Tyson, Jeff. (2013). Comparing DSL types. Retrieved May 30, 2013, from <http://www.howstuffworks.com/vdsl3.htm>.
- U.S. public libraries weather the storm. (2012). Retrieved February 2, 2014, from <http://www.ala.org/research/sites/ala.org.research/files/content/initiatives/plftas/issuesbriefs/issuebrief-weatherstorm.pdf>.
- Welcome to fiber cities. (2013). Retrieved June 4, 2013, from <https://fiber.google.com/cities/#header-check>.
- Who provides Internet service for my Internet service provider? (2013). Retrieved May 29, 2013, from <http://www.howtogeek.com/123599/who-provides-internet-service-for-my-internet-service-provider/>
- Wong, Q. (2008, May 1). WiMax to widen Net access: New device more powerful, secure than Wi-Fi. *Lexington Herald-Leader*, p. A3.
- Wrolstad, J. (2004, September 10). FCC: Broadband usage has tripled. Retrieved May, 2007, from http://www.newsfactor.com/story.xhtml?story_title=FCC—Broadband-Usage-Has-Tripled&story_id=26876.

Internet Technologies: TCP/IP

The core technology of the Internet is packet switching, implemented through the TCP/IP data communication protocols. Because all the higher-level protocols and many information services depend on this foundation, a detailed understanding of TCP/IP is an important goal of this text. Packet formation, packet addressing, MAC addresses, routing issues, subnet masks, address classes, IP address assignment, and the DNS lookups are all essential parts of TCP/IP. IPv6 has been an option for some time; however, for a variety of reasons, many organizations are delaying the transition from IPv4 to IPv6. Because IPv4 is still in common use, much of this chapter focuses on it, but key differences between IPv4 and IPv6 are highlighted throughout. There are a number of excellent books devoted entirely to TCP/IP including those by Fall and Stevens (2012), Kozierok (2005), and the older two-volume set by Comer and Stevens (1991a, 1991b). Other excellent sources are listed in the Additional Reading section at the end of this chapter.

PACKET SWITCHING AND TCP/IP

The ARPANET was the first packet-switching computer network, and the modern Internet developed from that simple beginning. Packet switching, introduced in Chapter 3, is the foundation of computer data communication; Leonard Kleinrock's (1996) early work on data communications with packet switching made ARPANET, as well as the Internet of today, possible. To review, packet switching is the process by which data are broken up into discreet "chunks," encapsulated in a digital envelope, and sent out over a network for delivery according to the rules of some addressing scheme (Krol, 1994). These packets can be mixed together with other unrelated packets in a process referred to as *multiplexing*; when they are separated from each other on the

70 Internet Technologies and Information Services

receiving end, it is called *demultiplexing*. The TCP/IP is the foundation packet-switching protocol of the Internet, and all the higher-level client–server protocols are included in the TCP/IP family of services.

Vinton Cerf and Robert Kahn were among the group that was instrumental in designing the original ARPANET; they went on to develop the TCP/IP network standards in 1980 (Leiner et al., 1997). ARPANET adopted the TCP/IP protocol in 1983, establishing the Internet of today (Zakon, 2011). The TCP/IP Internet has been described as a “dumb network with smart end hosts” because of two important guiding principles: the *end-to-end argument* and *fate sharing*. Together, these place responsibility for some network functions and maintaining connections on the nodes instead of lower network levels (Fall & Stevens, 2012). In simple terms, these two communication protocols, and the end point applications associated with them, are responsible for breaking up all the data that move on the Internet into small packets, encapsulating it into electronic envelopes, and applying a standardized source and destination address to facilitate its movement across the globe. The Internet is truly an open system; the TCP/IP standards and their implementations are publicly available with little or no cost. Understanding the basics of how TCP/IP works is central to understanding all Internet technologies and services.

TCP/IP represents two companion protocols working together. The IP part is responsible for the application of the source and destination address to all packets; each IP packet typically holds a TCP or, sometimes, a User Datagram Protocol (UDP, described later) packet. The TCP part is the process that takes the large piece of data and breaks it up into many small pieces, usually from 1 to about 500 bytes each. All data are first encapsulated into TCP or UDP packets, which in turn are encapsulated into IP packets, which are the packets that are routed on the Internet. The successful delivery of all the packets depends on locating the host with the destination IP address. The IP packets are delivered to the Internet host, and then the TCP takes over to reassemble the information into a coherent whole.

Krol (1994) explains packet switching with the analogy of how letters move through the U.S. mail. When a letter is sent via the postal service, the information is put into an envelope with a “To” and “From” address, as well as postage, all applied according to standardized rules. Then this packet is put into a mailbox with lots of other, unrelated packets. The successful delivery of the letter is analogous to the role of the IP protocol. However, this protocol is both “unreliable and connectionless”; it is “unreliable” because there is no guarantee of packet delivery nor is there any acknowledgement of receipt of the IP packet. It is “connectionless” because there is no ongoing

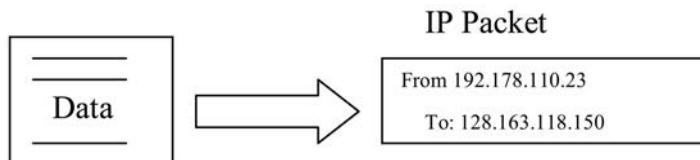


Figure 4.1 A simplified representation of an IP packet.

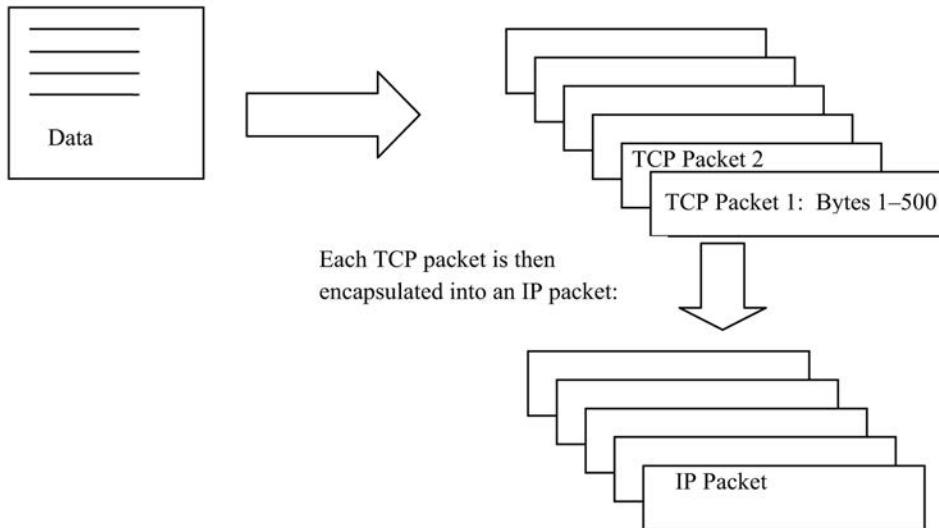


Figure 4.2 A representation of how TCP is combined with IP. Data is broken up into multiple TCP packets, each of which is placed in an IP packet and sent over the network.

connection between the source and destination hosts. Figure 4.1 represents an IP packet.

To extend this analogy to TCP, assume an additional rule is imposed; namely that each envelope can only carry a single sheet of standard letter paper. To send a 10-page letter, the pages would need to be numbered and put into separate envelopes, each of which is then addressed and mailed. The delivery of each “letter” is the job of IP. However, the receiver of the multiple letters must now open each and reassemble the pages in the intended order, which is analogous to the job of TCP. In addition, TCP packets carry a port assignment to designate which client program they are destined for at that host. Continuing with the mail analogy, consider mail to an apartment building; a street address needs to be refined with the unit number. The job of IP is to get the packets to the right host; TCP uses port assignments to ensure they get to the right client program on that host. Figure 4.2 represents the idea of how these two protocols work together.

IP PACKET ADDRESS HEADERS

This packet technology has significant overhead, that is, a significant portion of the packet itself is not data, but instead is information needed to identify and move the data. The kind of overhead that is present in network packets has some similarities to the overhead in another standard that librarians are familiar with—the MARC record. The MARC record was designed to facilitate the transmission of “machine readable cataloging” (i.e., bibliographic data in electronic form). This record format has a leader and directory containing

72 Internet Technologies and Information Services

meta-information useful to the reliable transmission of the cataloging record. Similarly, the fields in the IP and TCP packet address headers contain the meta-information needed to get the packets to the correct destination and check for possible errors. The structure of the IPv4 address header is shown in Figure 4.3.

The packet header identifies the IP version (IPv4 is still a common standard, but IPv6 is gaining acceptance), the header length, the length of the packet itself, and the service type, which associates differing levels of service priorities for packets using a unique packet identifier. The fields that are labeled **flags**, **fragment offset**, **header checksum**, and **options** determine how this packet fits into a larger fragmented message (fragment offset), performs header error checking (checksum), and provides information about routing (options). The **TTL** field is the “time to live” and specifies the finite lifetime of an IP packet; without this, undeliverable packets could be clogging the Internet forever. The most critical fields are those for **source address**, **destination address**, and **data**. The header fields for IPv4 addresses are 32 bits long. The data field is typically a TCP or UDP packet. An IP packet carrying a TCP packet is an example of *packet encapsulation*. Encapsulation can also involve higher levels of nesting, for instance when the data portion of an Ethernet frame is an IP packet, which in turn carries a TCP packet.

The IPv6 datagram is functionally similar in that it contains addressing information and encapsulates data in the form of TCP or UDP packets. However,

Bit :				31				
0	4	8	16					
Version	Length	Type of Service	Total Length					
Identification			Flags	Fragment Offset				
TTL	Protocol		Header Checksum					
Source Address								
Destination Address								
Options								
---Data ---								

Figure 4.3 The IPv4 address header.

it has a slightly different structure and added features. It has multiple header areas—a main or base header for addressing and extension headers for added information if needed as shown in Figure 4.4. The checksum field was eliminated and a “Flow Label” field was added that can be used to prioritize packets (Kozierok, 2005). The base header allocates 40 bytes for all this overhead; 16 of these bytes (128 bits) are set aside to accommodate the longer source and destination addresses of IPv6. The structure of this 40 byte IPv6 base header is shown in Figure 4.5.

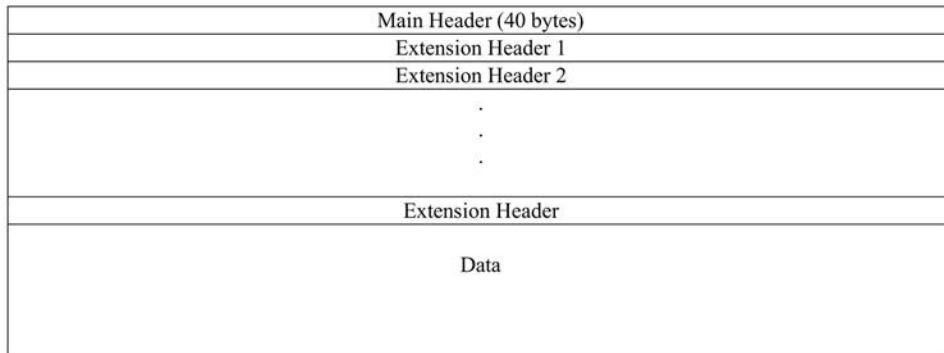


Figure 4.4 The IPv6 datagram.

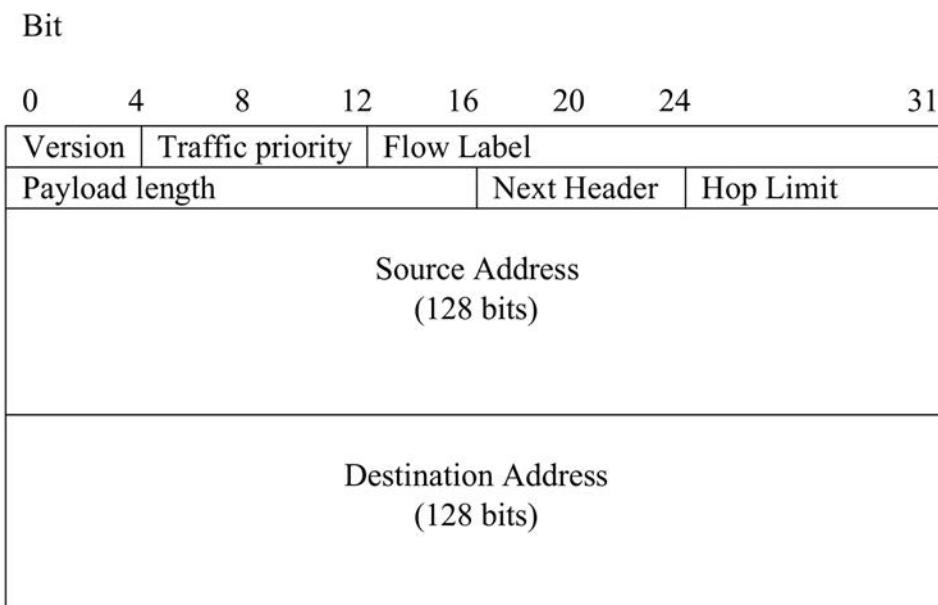


Figure 4.5 The IPv6 address header.

ARP

ARP (*Address Resolution Protocol*) is the basic protocol of TCP/IPv4 that manages the correspondence between network and hardware addresses. Although the newer IPv6, discussed later in this chapter, uses the Neighbor Discovery Protocol (NDP) and not ARP, most IP traffic still depends on IPv4 and ARP. A key idea of the Internet was to interconnect many different kinds of devices and networks, each of which has different addressing schemes that identify a node. In addition, device addressing is controlled by the manufacturer and IP addresses are dependent on the address pool available to an organization and are dynamically assigned. Thus a device always has the same hardware address called a MAC address but the IP address associated with it can vary from place to place and over time. The most common interaction managed by ARP is between 48 bit Ethernet type addresses and 32 bit IPv4. ARP uses broadcast messages over the network to manage this process. Examples of how this correspondence is managed are given in the section on assigning IP addresses.

As with all TCP/IP protocols, ARP sends packets to send and receive these address requests. The ARP frame header contains information about the length and type of the request/message fields that follow. The request (or reply) message again includes the source and destination hardware address as well as the sender's and target IP address. To make the process more efficient, the ARP protocol caches the results of these exchanges. The **arp** command shows the contents of this cache.

TCP, UDP, AND ICMP

TCP is a reliable protocol designed to ensure all the “chunks” of data are delivered and reconstituted into a coherent whole. There is significant overhead associated with TCP headers, as shown in Figure 4.6.

It is not necessary to this discussion to examine the details of each field in this header, but a few important points about TCP header information are worth a closer look. First, there are 16 bits reserved for port assignments. Ports, described in Chapter 3, are a communication channel for network data communications. All the various higher-level Internet protocols such as telnet and HTTP have default port assignments. Because TCP/IP uses 16 bits to identify a *port*, there are 65,534 ($2^{16} - 2$) possible ports (2 is subtracted from this total because binary addresses that are all zero or all ones are not permitted as port numbers). Other key header components include the sequence number, which allows the “chunks” of data to be reassembled, and the acknowledgement field, which confirms the receipt of the data and makes this a “reliable” protocol. Missing packets are re-requested, so TCP does care about the connection between the hosts, at least as far as confirming packet delivery. The TCP acknowledgement feature involves a client synchronization (SYN) request, a server response to it (SYN-ACK), and a final client acknowledgement of that server reply; this process is referred to as the TCP “three-way handshake.” As discussed in Chapter 6, this feature can be exploited to perform a denial of service attack.

Bit :

0

16

31

Source port	Destination port
Sequence number	
Acknowledgement number	
Data offset, Reserved area, etc.	Window
Checksum	Urgent pointer
Options	
--- Data ---	

Figure 4.6 The TCP address header.

The UDP is an alternate to the TCP protocol. As described previously, TCP has significant built-in error checking to ensure the integrity of the delivered data and to create a connection-aware environment until the successful delivery of all needed data. However, there are times when this added reliability is not important and it adds unnecessary overhead to the process. For instance, audio and video streaming use UDP. With streaming, a player connects to a server, determines the bandwidth of the connection, and starts to play as soon as the incoming packet stream can keep up with the output of the music or video. These applications give priority to the speed of packet formation and delivery, and the acknowledgement features of TCP would just slow the process down. UDP, shown in Figure 4.7, has less overhead than TCP, and it is better suited to streaming technologies because by the time a missing packet could be resent, the stream would already be past the place where it was needed.

The Internet Control Message Protocol (ICMP) is used for error checking between gateways and hosts and between different hosts at the Internet layer. It handles error reporting and plays a role in routing decisions. ICMP is another type of datagram carried in an IP packet just as UDP or TCP packets are. There are helpful utilities used for network testing and troubleshooting that

Bit:

0

31

Source Port	Destination Port
Length	Checksum
--- Data ---	

Figure 4.7 The UDP packet header.

are based on ICMP. However, ICMP can create security issues and network administrators often block these messages with firewalls, which not only provides some protection but also prevents the use of diagnostic utilities such as ping or traceroute.

ICMPv6 extends the role of this protocol by providing information about router and host configuration in addition to informational error messages. IPv6 does not support broadcast messaging; it relies instead on the ICMPv6 protocol to utilize multicast messages for addressing rather than the ARP broadcast messages that are used for this function in IPv4. IPv6 uses the NDP in place of ARP to manage address mapping (Fall & Stevens, 2012).

Everything that takes place via the Internet, whether it is sending or receiving email, using VoIP, browsing the Web, listening to music, or watching a video, depends on these packet technologies. The next time you are impatient with a slowly loading Web page or halted video stream, consider all the packets that must be created, addressed, delivered, and reassembled to present the content you see.

IP ADDRESSING

IP addressing is the foundation for all Internet technologies. Network management depends on a thorough understanding of IPv4 address structure, IP address classes, and the strategies employed to administer and assign IP addresses within an organization. Many information services use IP addresses or an IP proxy as a mechanism to manage licensing agreements that require user authentication. This discussion of IP addressing focuses mostly on IPv4.

Computers depend on binary numbers and do not share the human bias for base 10 numbering. Success with this section depends on familiarity with numbering systems, specifically base 2 or binary numbers, hexadecimal numbers (base 16), and the familiar decimal numbers (base 10). IPv4 uses 32-bit numbers for the source and destination address. These addresses are in the form of 32-bit

binary numbers; a series of 32 zeros and ones are used to represent the pool of unique addresses. Assuming all these addresses are available, there are at most 2^{32} possible IPv4 addresses, or approximately 4.3 billion. However, there are restrictions that reduce this potential pool of addresses associated with each address class. This pool of IP addresses seemed more than adequate in 1980 when TCP/IP was developed; after all, in 1981, there were only about 200 hosts on the Internet. However, the demand for IP addresses has stretched this original scheme, as more devices need IP assignments. Users may need multiple IP addresses to support many devices such as a desktop computer, a laptop, mobile phone, or other device. As discussed in Chapter 1, the “Internet of everything” will likely result in a need for many more IP addresses. Organizations have become increasingly Internet connected; for instance, universities no longer just connect offices, but computer labs, dorm rooms, libraries, and public areas with ports for laptop connections. The high demand for IP connectivity has strained the existing addressing system. There are a number of workarounds for this problem such as using the NAT protocol and VPNs to extend the address space. Migration to IPv6, a newer version of IP that uses 128-bit numbers for addressing, is another solution. The address pool for IPv6 is 2^{128} —a very big number.

The 32-bit numbers of IPv4 are expressed as a set of four octets. Computers use these in binary form, but when written for humans they are shown in decimal notation, each separated by a period and referred to as a “dotted quad.” Because each part of this is an octet of binary numbers, the possible decimal values are between 0 and 255 because $2^8 = 256$ possible values, starting with 0. An example of a dotted quad could look like this: 128.163.188.75.

The set of possible addresses is divided into classes for administrative purposes, cleverly named Class A, Class B, Class C, Class D, and Class E. This discussion focuses on Classes A–C; Classes D and E do not really factor into this analysis (Class D is used for special nodes and multicast groups, and Class E is reserved for future use). IPv6 does away with this class structure and instead of the dotted quad notation divided among 8-bit boundaries of IPv4, the longer 128-bit addresses of IPv6 are expressed in colon hexadecimal notation. In this format, the 128 bits are divided along 16-bit boundaries, each represented with four hexadecimal numbers, as shown in the example below:

21EA:00D3:0000:2D3B:02CA:00FF:FE28:9B5A

This format can be further compacted by removing leading zeros within each block, but each block must contain at least one digit.

There are two facets to understanding the IPv4 addressing scheme: first, classes assign different octets for the identification of the network and hosts on that network; and second, there are restrictions on the values for the first octet of the network identifier. For Class A networks, one octet is reserved for network ID, and three octets for hosts; for Class B, two octets are reserved for network ID, and two octets for hosts; and for Class C, three octets are reserved for network ID, and one octet for hosts as shown in Figure 4.8.

It is apparent from this structure that there cannot be very many Class A networks, but each one can theoretically have a large number of hosts (2^{24} or $256 \times 256 \times 256$ or about 16 million). Because values of all zeros or all ones are not permitted in either the network or host portions of the address this would really be $2^{24} - 2$. A Class B network could only accommodate $2^{16} - 2$ or 65,534 hosts, a Class C can have only $2^8 - 2$ or 254 hosts.

78 Internet Technologies and Information Services

Class A	N	H	H	H
Class B	N	N	H	H
Class C	N	N	N	H

Figure 4.8 The three most common IPv4 address classes.

The added address constraints relate to permitted values for the first octet and the fact that the host portion cannot be all zeros or all ones in any class. The first octet restrictions are: Class A, the high-order bit (the bit on the far left of the eight-digit binary number), must be 0; for Class B, the high-order bits must be 1 and 0; and for Class C, the high-order bits must be 1, 1, and 0. Therefore, Class A addresses can have a first octet value from 1 to 126 (the 127 value, while it appears to be available, is reserved for special purposes), Class B networks can have a first octet value from 128 to 191, and Class C networks must begin with a number in the range of 192–223. The permitted decimal values and how they are obtained are shown in Figure 4.9.

A local network based on TCP/IP is an *intranet*. Intranets can use any of the class addresses allowed in TCP/IP. However, when an intranet connects to the Internet it must use the appropriate class designation assigned to it from the domain registration. As can be seen from how these classes are structured, the type and size of the organization determine the appropriate class assignment. Figure 4.10 summarizes the networks and hosts possible within each class.

Class	Maximum Binary Value of First Octet	Decimal Equivalent
A	0 1 1 1 1 1 1 1	127
B	1 0 1 1 1 1 1 1	191
C	1 1 0 1 1 1 1 1	223

Figure 4.9 Constraints on the first octet for Classes A, B, and C.

Class	# Networks	# Hosts
A	126	16,777,214
B	16,256	65,534
C	2,064,512	254

Figure 4.10 Number of networks and hosts for Class A, B, and C.

A useful special address is 127.0.0.1, known as the *loopback address*. This diagnostic address redirects packets back to the originating host thereby testing the TCP/IP installation on the host machine.

Private IP Addresses, NAT, and APIPA

One way for organizations to extend their IP address space utilizes private, internal IP addresses assigned to a single public IP. The usual address ranges reserved for this function within each IP class are shown in Figure 4.11, but organizations are free to manage internal addressing in any way they choose. For example, my institution is a Class B address space but although IP addresses for student dorms begin with the expected 172.xx, many offices have private addresses that start with 10.168, which is in the private Class A range.

The NAT protocol manages the correspondence between the public and private addresses. This works much like a proxy server by serving as an intermediary for the packet streams. Packets going outside the network get a public IP assigned to them that is associated with this proxy device but use the private IP address when staying within the internal network, expanding the address space available to organizations and ISPs. For instance, nodes in a campus dormitory usually use private network addresses. Most ISPs handle the IP assigned to customers in this fashion as well. Sites such as www.whatismyip.org can reveal the public IP address that is associated with that private address by viewing the HTTP header information. NAT is supported by almost all routers and its use has expanded as organizations have needed more IP address space. The success of NAT has slowed the migration to IPv6 because it is a low-cost way to expand the number of nodes on a network using a much smaller pool of routable IP addresses.

APIPA (Automatic Private IP Addressing) is an autoconfiguration strategy that generates a dynamic private IP address if no other address has been manually configured and no Dynamic Host Configuration Protocol (DHCP) server can be found (DHCP is discussed later in this chapter). The private address range of 169.254.1.1 through 169.254.255.254 is reserved for this purpose and serves as a self-configured LAN address until a DHCP server responds with an IP address, which continues to be sought every few minutes. Microsoft has enabled APIPA in Windows and this type of private address will thus be reported by a Windows machine even when it is unconnected from a network.

IP Class	Private IP address ranges
Class A	10.0.0.0 – 10.255.255.255
Class B	172.16.0.0 – 172.16.255.255
Class C	192.168.0.0 – 192.168.255.255

Figure 4.11 Internal IP address ranges for each IP class.

IPv6

Even though an IP address space can be extended with private network addresses as described previously, IPv4 is reaching its limits. The limitations of this address space have been recognized for some time, and by the early 1990s, proposals for the Next Generation IP (IPng) were developed. IPv6 was derived from one proposal known as Simple Internet Protocol Plus (SIPP) presented in RFC 1710, an Internet Engineering Task Force “Request for Comments” white paper (Simoneau, 1997). One goal of IPv6 is to solve the IP address space shortage, but it is also an opportunity to simplify the IP header and enhance the privacy and security of data transmission. IPv6 abandons the address “class” structure of IPv4 and instead identifies three types of addresses: unicast, multicast, and anycast. For instance, a multicast sends packets from a single host to multiple addresses simultaneously (Loshin, 1999). Beyond addressing and security, IPv6 is better for audio and video data. It is also more flexible than IPv4 because not all protocol features are specified.

The Internet Society dubbed June 6, 2012, as the world launch day of IPv6. The acceptance of IPv6 is dependent on strategies ensuring backward compatibility with existing IPv4 networks. However, even though its use has doubled each year for the last several years, there is still somewhat of a standoff among the players who would implement it—ISPs see little gain for the costs and others may be not ready for IPv6 (Fiocco, 2013). Most new network hardware is compliant, but many legacy systems are not. It is estimated that about 56 percent of ISPs in the United States do not currently support IPv6 (Coffeen, 2013). At my university, although the network is ready for IPv6, a service such as the IP tables in the Blackboard LMS version in use is not compatible with IPv6 addresses. In addition, there are alternate strategies; NAT along with the security of RFC1918 solves many of IPv4’s immediate problems. The bottom line is that although IPv6 offers advantages, deployment continues to be slow because many network administrators do not feel much urgency to fix a problem that does not yet seem critical and that is expected to have significant costs (Kieran, 2005). According to reports from the North American IPv6 Summit, only 2.5 percent of U.S. Internet traffic uses IPv6 as of 2013. For those transitioning to IPv6, there are two approaches that allow its implementation in what is still an IPv4 world. One is to provide dual support for IPv4 and IPv6 in all hosts and routers, and the other is to permit the “tunneling” of IPv6 packets within IPv4 headers (Loshin, 1999; Simoneau, 1997).

Although it appears market forces alone have not yet incentivized wide scale IPv6 adoption, there will be increasing pressure to do so; it is predicted that based on the average demand for IPv4 addresses of about 200 million per year, there could be a shortfall of about 1 billion addresses by 2014 (Coffeen, 2013). Longer term, it is likely that such pressure will create a tipping point in favor of IPv6 adoption.

Managing IP Address Assignments

For a host to be on the Internet, it must be assigned an IP address. There are different strategies for the management of this assignment. In the early days

of Internet use, most users connected via a shell account, that is, an account on a large-scale computer such as a mainframe that was the host connected to the Internet. This required that a non-Internet dial-up connection be first established between the PC and the mainframe host; the mainframe then served as the jumping off point to the Internet. Early ISPs such as CompuServ and America Online offered direct Internet access via Serial Line Internet Protocol/Point-to-Point Protocol (SLIP/PPP), making the local PC an Internet host. ISPs provide a variety of connection choices in almost all locations. Many ISPs assign addresses with a newer variation called *PPP-over-Ethernet* (PPPoE); this is used when the DSL modem acts as a bridge or switch rather than a router when it is connected to a PC with an Ethernet cable (Fall & Stevens, 2012). Other options are connections to the Internet via an Ethernet network or Wi-Fi. Several events in the 1990s made it much easier to get direct access to the Internet, beginning with the introduction of Windows 95 when TCP/IP became part of the OS. This was a huge leap forward because earlier versions of client software for Windows required a separate WINSOCK (Windows sockets) installation to direct the TCP/IP packet stream to Windows client applications. Integrating this function into the OS eliminated the problem of multiple, and often incompatible, WINSOCK versions. The proliferation of ISPs using SLIP/PPP provided IP addresses assigned directly to the desktop host. SLIP was the initial approach for these dial-up connections, but SLIP was not really a standard. The improved and more efficient PPP and the PPPoE standards, with better error checking and compression technology have replaced SLIP.

The direct Ethernet-based connection is typical for Internet access within larger organizations and companies. In this scenario, an appropriate IP address is assigned to the corresponding EA making it an Internet host. Strategies for handling this IP assignment included manually hard coding an IP address in the PC setup, the Reverse Address Resolution Protocol (RARP), the Bootstrap Protocol (BOOTP), and the DHCP. Entering the desired IP address manually sounds like the simplest way to make this assignment, but this is problematic for many reasons; however, it can be useful as a way to troubleshoot connection problems. For example, a system that is failing to connect can be tested by substituting another address known to be functioning on the network. In Windows, the TCP/IP properties are available through the control panel Network Connections icon.

RARP, BOOTP, PPPoE, and DHCP are all ways to manage and possibly automate the assignment of IP addresses, but almost all networks now use DHCP to automate this assignment while the RARP and BOOTP approaches have become relatively rare and obsolete. The hardware address for the Ethernet port, known as the MAC address (also referred to as the physical address or EA), is hard coded with the hardware. However, IP addresses are logical addresses and come from different classes depending on the type of the organization. The assigned address space comes from the domain registration process. Once assigned, it is up to the local administrators to manage the address space. The protocols designed to exchange Ethernet and IP address information are ARP and RARP. ARP, discussed earlier in this chapter, allows routers to request that the host with a certain IP address respond with the associated MAC address. Why this is done is discussed in the section on routing. RARP does the opposite, that is, it requests the IP address associated

with some known MAC address. This simple protocol can be used to request an IP address for a MAC address on a network. However, RARP by itself had limitations; for instance, it did not allow the use of subnet masks. The BOOTP approach was similar but had been refined to include other parameters such as the subnet mask as well as the IP address. In BOOTP, the computer booting up on the network broadcasts a request message with its MAC address. The BOOTP server receives the message and responds with the IP address as well as other network configuration information as shown in Figure 4.12. In this approach, network administrators maintain a database of registered MAC addresses and associated IPs on the BOOTP server. This usually results in the same IP address assignment for that host each time it boots on the network. Separate from assigning the IP, the database also helps with network administration by tracking the computer hostname, location, and the contact information of the user. Registration of new equipment on the network is required for IP assignment with BOOTP.

The main approaches to IP address management for most LANs or WANs are now DHCP or PPPoE for many home networks. DHCP looks very similar to BOOTP, and it also depends on the exchange of broadcast messages between the host and some server. However, there are some key differences with BOOTP. With DHCP, there is a pool of available IP addresses, and one of these is dynamically assigned to the host for that session (see Figure 4.13). The IP address is “leased” for a period by that host, but once it expires or releases, a different address might be assigned for future sessions. Having a pool of dynamic addresses available is useful for a number of reasons; for example, users or guests can connect a laptop or a new desktop computer to the network and immediately gain Internet access without being previously registered on the network.

As described previously, if an IPv4 host tries to connect to a network without a DHCP server, an IP address can be generated automatically using an APIPA generated link-local private address. IPv6 also has an autoconfiguration feature that can allow nodes to self-assign a link-local IPv6 address automatically, which requires little or no manual configuration of hosts (Walls, 2006). IPv6 ensure that such an address is available with the Duplicate Address Detection (DAD) feature, which utilizes Neighbor Solicitation and Neighbor Advertisement messages to share this information.

IPv6 uses DHCPv6, which is conceptually very similar to DHCPv4 and serves the same role in assigning addresses. However, the IPv6 address’s lifecycle is

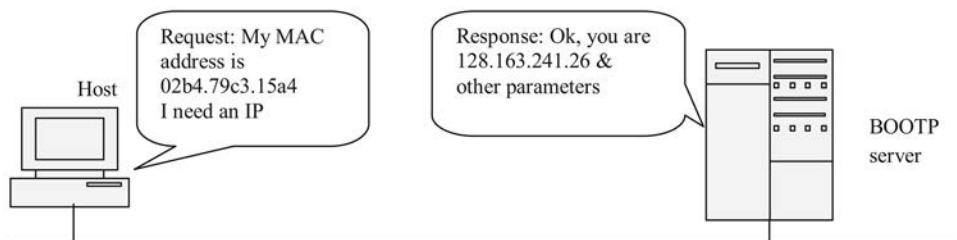


Figure 4.12 BOOTP address assignment.

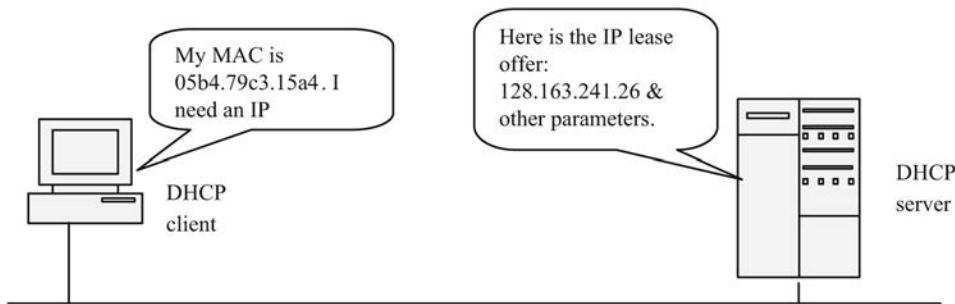


Figure 4.13 DHCP request and lease offer.

more complex than that of IPv4. Hosts using IPv6 have multiple addresses that are valid for specific times and purposes; these addresses have states that can thus change over time. Instead of ARP broadcast messages, IPv6 uses the Neighbor Discovery Protocol to handle address mapping. This protocol allows nodes on the same network segment to find each other and determine what type of connectivity they have. It also supports stateless address autoconfiguration. The address mapping and router discovery are handled by the Neighbor Solicitation/Advertisement parts of this protocol (Fall & Stevens, 2012).

For home networks, PPPoE is often used by the ISP instead of DHCP because it provides additional configuration options and audit logs when the DSL modem is used as a switch instead of a router. In this scenario, the device connected to the modem, usually a PC or wireless router manages the IP routing and addressing functions (Fall & Stevens, 2012). One possible complication of dynamic addressing is that the IP assigned is not always the same, so it cannot be used for computers running a service such as a Web server that requires an unchanging IP assignment to associate with the server name.

ROUTING AND SUBNET MASKS

Routers are essential devices for the movement of all Internet packets and they determine the appropriate path packets should take. Routers use IP addresses, and they “talk” to the devices that are connected to it to learn their IP addresses. The addresses of all the devices attached to the router are then stored in the router table. The initial routing decision is based on the answer to a simple question: is the packet destined for a host on the router subnet or to a host in the outside world? To make this determination, the router applies the *subnet mask*. A mask refers to the idea of hiding something to isolate one element of interest (think of “masking” tape used when painting windows). The examples in this discussion of routing and subnets are all based on IPv4.

The management of an IPv4 address space often utilizes *subnets*, which are created by “borrowing” some of the bits reserved for host ID to extend the

84 Internet Technologies and Information Services

network ID portion. Although it is possible to “borrow” any number of bits from the host octets, the simplest example is when a full octet is used for this purpose; this is a *byte boundary* subnet mask. Subnets are used when different network protocols connect at a router. They also perform a number of other useful functions:

- Subnets keep the pool of network addresses smaller and therefore they are easier to manage and troubleshoot.
- They reduce overall network traffic because packets are often destined for hosts on the same subnet.
- They make routing tables smaller and more efficient.
- They allow for geographical association of network addresses within buildings or areas.
- They make network security easier to manage.

Each IPv4 address class has a default subnet mask. For a Class A, it is 255.0.0.0; for Class B, 255.255.0.0; and for Class C, 255.255.255.0. For example, consider a Class B byte boundary mask. There are two octets associated with the host ID, which means there can be $65,534 (2^{16} - 2)$ possible hosts to manage on such a network; two is subtracted because addresses that are all 0s or all 1s are not permitted. Having all these addresses together as one network is more difficult to manage and results in less efficient packet routing because each router must maintain a very large routing table with all these addresses. For a Class B network, the byte boundary mask functionally results in three octets associated with the network part of the address; the two octets assigned to all Class B addresses by default, along with the first octet of the host portion of the address. This third octet becomes the subnet assignment; making it possible to have $254 (2^8 - 2)$, again removing the two addresses with all 0s and all 1s) subnets, each of which can have $254 (2^8 - 2)$ hosts. This partitions the network.

As mentioned previously, the number of borrowed bits can be less than a full byte. The number of bits borrowed is determined by the number of subnets needed and the number of nodes expected on each subnet. The general formula for calculating subnets is $(2^n) - 2 =$ the number of subnets or hosts, where “ n ” is equal to the number of bits in the mask. So, if the number of bits in the mask is 3, then $2^3 - 2 = 8 - 2 = 6$ subnets.

All IP address classes can utilize this strategy and borrow differing numbers of host bits to accomplish the needed configuration. Simoneau (1997) describes the subnet tables for Class A, B, and C networks. Note that for a Class C network, borrowing a full byte is not an option, as that leaves no bits for host addressing. The process of creating a partial byte subnet mask and the resulting number of subnets and hosts is illustrated in the Class C subnet table in Figure 4.14.

The application of a default or modified subnet mask provides a simple process to separate the network part of the address from the host part so routers can determine if packets are to stay within the local network or to go outside of it. To illustrate the application of a subnet mask, the default mask for a

Subnet Bits	Subnet Mask Value	Last Octet in Binary	Number of Subnets	Hosts
0	255.255.255.0	0000 0000	0	254
2	255.255.255.192	1100 0000	2	62
3	255.255.255.224	1110 0000	6	30
4	255.255.255.240	1111 0000	14	14
5	255.255.255.248	1111 1000	30	6
6	255.255.255.252	1111 1100	62	2

Figure 4.14 Class C subnet mask table—the number of subnets or hosts is based on formula $2^n = \text{total}$, where n equals the number of bits associated with mask or host (Simoneau, 1997).

Class B network is given to reveal how it enables a router to make the routing decision.

Assume a Class B network has the domain address 128.163 and a default subnet mask of 255.255.0.0. If a packet is sent between two machines on the same network the network portions of the packet addresses are the same; for example, assume Machine A has the address 128.163.118.27 and Machine B the address 128.163.118.25. (Remember these addresses are shown in decimal notation for our convenience, but the machine addressing is in binary.)

The router sees the source and destination addresses in the IP packet header. If the network portion of both addresses is the same, the packet is destined for a host on this network. To determine this, the router applies a Boolean AND with the subnet mask and the IP address associated with the packet source, as shown in Figure 4.15. (Note: Boolean logic is discussed in other areas of this text; for the purposes of this discussion, the readers need to know that with AND, the only time the result is 1 is when both operands are also 1.)

To determine the network part of the ID for the destination IP address, the same operation occurs for address B, as shown in Figure 4.16.

The network part of the ID is preserved for both addresses; this is the only possible outcome of a Boolean AND performed when the mask values are 1s.

Machine A:	10000000.10100011.01110110.00011011
AND	
Mask:	11111111.11111111.00000000.00000000
Result:	10000000.10100011.00000000.00000000

Figure 4.15 Boolean AND with address A and subnet mask.

Machine B:	10000000.10100011.01110110.00011001
AND	
Mask:	11111111.11111111.00000000.00000000
<hr/> Result:	10000000.10100011.00000000.00000000

Figure 4.16 Boolean AND with address B and subnet mask.

Machine A:	10000000.10100011.01110110.00011011
AND	
Mask:	00000000.00000000.11111111.11111111
<hr/> Result:	00000000.00000000.01110110.00011011

Figure 4.17 Boolean AND with address A and the inverted subnet mask.

The first two octets are the same for both addresses so the router determines the packet is destined for a machine on the same network.

Subnet masks permit routers to isolate the host portion of the address when needed by first inverting the value of the mask and again performing a Boolean AND with the addresses. This is shown for address A and the inverted subnet mask in Figure 4.17.

This process preserves the host part of the ID when the router needs the host portion. The overall result is that routers use this simple Boolean approach with the subnet mask value to make a routing decision. If the network parts of the “from” and “to” address are the same, the packet is destined for a host on the same network. If this is the case, the router lets the Ethernet frame carry the traffic directly to that host. However, routers only see IP addresses, and Ethernet needs a MAC address, so the router gets the needed MAC address by way of ARP. The router broadcasts a message on the network asking for the machine with the IP address in question to respond with its corresponding MAC address; this MAC address is then used to move the data to the destination by way of the Ethernet.

If the application of the subnet mask determines the network portions of the addresses are different, that means the packet must be destined for a host on a different network. Getting the packet on its journey to the outside host is the job of the default gateway machine. The default gateway is ARPed, the data

are sent there as an Ethernet frame, and then the default gateway determines where the IP packet should go next in the outside world.

FIREWALLS AND PROXY SERVERS

Firewalls protect networks by dropping certain packets and preventing them from entering or leaving the network boundary. This isolation can prevent unauthorized access and directed attacks or malicious content from getting into the network as well as stopping outgoing traffic to specific outside sites. This can be accomplished either by packet filtering routers or with proxy servers. Firewalls that act as routers are quite common; they can forward or drop packets with headers that meet, or do not meet, certain criteria. This is accomplished by the network administrator via an access control list; this might include specific Internet addresses, certain types of ICMP messages such as those associated with a ping request, or TCP/UDP services associated with specific ports.

Proxy servers were introduced in Chapter 3 and they are relevant to this discussion. Proxy servers have many uses with TCP/IP networks: they can serve as firewalls, to authenticate users to a service, and to restrict or filter Internet access. The term *proxy* is quite literal; it means another host stands in as your proxy for Internet activities. Proxy servers allow for higher security, and K-12 schools often use them to centralize control of filtering activities. Many libraries provide services requiring authentication of the user before access to the service or database is granted. Often such authentication is transparent to the user through IP authentication. At my university, most IP addresses begin with the Class B domain assignment of 128.163. License-restricted resources are accessed without a separate login through automatic authentication based on the presence of a known domain address (or perhaps a specific address range within the domain) in the IP header of the request. For remote users, a proxy server acts as the intermediary between the end user and the licensed resource. The user is authenticated to the proxy server via a library or student identification number, and subsequent requests for service are forwarded from the proxy server with the approved IP. Users on the network are automatically authenticated by virtue of their source IP address, but off-network users must use either a proxy server or a VPN to present an approved IP address to the service.

THE DOMAIN NAME SYSTEM

Throughout this discussion, the emphasis has been on the numeric IP addresses used by TCP/IP and routers to move data. However, people do not typically use these numeric addresses when using the Internet; people prefer names. The domain name system (DNS) facilitates both the registration of domain names and the assignment of names to numbers. This information is then stored in servers to create a mechanism to automate the lookups that happen with each Web client request to go to a site. These functions are part of the DNS.

Domain Name Registration

The Network Information Centers (NIC) and the Network Service Centers (NSC), originally part of the NSFNET, were the official providers of Internet domain name registration services to the world, offering registration of domain names in the.COM, .NET, and .ORG top-level domains (TLDs). InterNic (<http://www.internic.net>) operated under the umbrella of the U.S. Department of Commerce and provided the public information regarding Internet domain name registration services. Then IANA took over central control for domains, addresses, and protocols. All this is now taken care by ICANN (<http://www.icann.org>), a nonprofit corporation that until recently operated under the U.S. Department of Commerce.

Domain names use a high-order suffix to identify either the type of organization or the country of the site. The common Internet domain suffixes include **com**, for commercial organizations, **edu** for educational organizations, **gov** for government entities, **mil** for military sites, **org** for other organizations, and **net** for network resources. In the late 1990s, ICANN added seven new domain names to join this list of generic TLDs (gTLDs) on the root server: **biz** and **info** for general purpose; **name** for personal; and four that are restricted to specific communities of sites: **museum**, **aero**, **coop**, and **pro**. In addition, two-letter country codes are also available. Examples of country codes include **us** for the United States; **au** for Australia; **fr** for France; **uk** for the United Kingdom; and **ie** for Ireland. In 2011, ICANN approved a huge expansion of the gTLD space when it announced it would accept applications for unique TLD names in place of the generic .com or .net suffix, potentially offering up to one thousand new top-level names per year. As of this writing, ICANN has received about 1,900 applications for suffix names; some are general such as .car and .book, others are for a company or organization such as .apple, or .aarp. Given the high application fee of \$185,000, most of the applicants are large companies.

For a time, buying and selling domain names was big business. In 1992, there were only about 5,000 registered domain names, but by the late 1990s, there were that many requested every day. By mid-2000, estimates had grown to 15–17 million registered names (Arnold, 2008). Domain name speculation was common; in the early days of the dot com bubble, some names were auctioned for millions of dollars. Examples include the sale of Business.com for \$7.5 million, Loans.com for \$3 million, Autos.com for \$2.2 million, and Bingo.com for \$1.1 million (White, 2000). There could be renewed interest in such transactions with the new TLDs being offered because applicants do not have to own a trademark to apply for a TLD. Once a new TLD is granted, the owner essentially becomes a registrar and can lease or sell access to it (Warren, 2011).

The process of registering a generic domain name is quite straightforward and begins with determining if the name you want is already in use. Registration information is stored in a database that can be queried by individuals using **whois** command utility or at the InterNIC website. Alternatively, there are many commercial domain registration services; Google, Yahoo, and many other companies can provide these services. The registration process is shown in Figure 4.18. Each TLD has an associated registry that feeds into a TLD server, as shown in Figure 4.19.

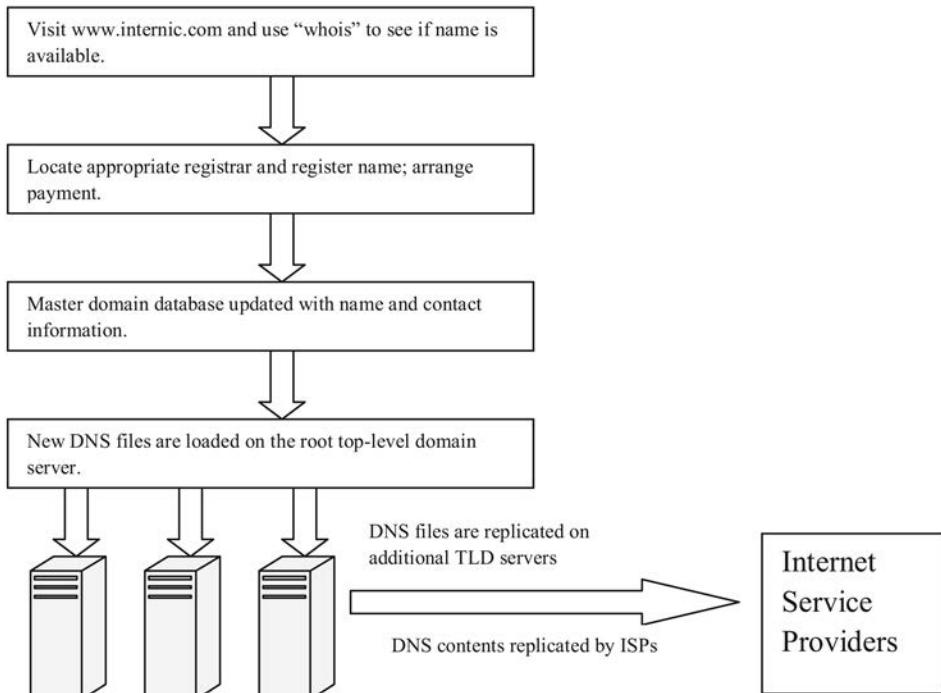


Figure 4.18 Registering a domain.

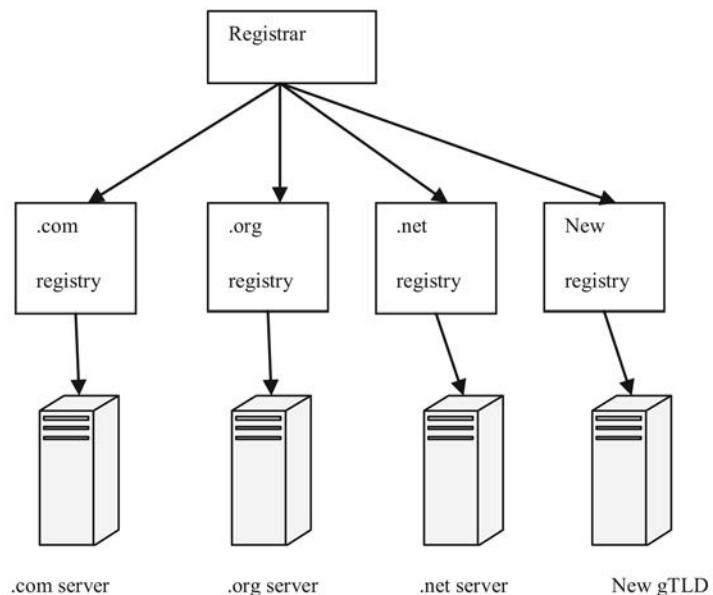


Figure 4.19 Registration information goes from the registrar to the registry for that TLD to the root server.

Each domain has dedicated servers to handle services for the registrants. The name associated with a machine at a domain is a FQDN. For example, the University of Kentucky is assigned the **uky.edu** domain name. There are a number of named servers at that domain, such as www.uky.edu (the main campus Web server) and sweb.uky.edu (the student Web server). When a Web client is used, the location entered is a FQDN with a filename, but TCP/IP needs the assigned IP address. The correspondence between the numeric IP and the registered name is managed by a *DNS lookup*.

DNS Lookups

The essential job of a DNS lookup is to convert the names people prefer for Internet domains and URLs into the number the protocol actually uses. When a URL is submitted or a link is followed, a request header is formulated by the client software, which is sent to the appropriate server. The server responds with a response header along with the content of the requested page. However, before the client request header is sent to the server, a lookup must happen to translate the URL into the IP number needed to address the packets of the request. If the URL **http://www.uky.edu/welcome.html** is entered in the location bar of a browser, the client formulates a request header to send to the server via TCP/IP. However, the packets of the request need the numeric IP address for that server. The DNS is a hierarchical system of servers maintaining a database of names and addresses providing these lookup services that associates an IP addresses with a URL. To support IPv6 addresses, a new DNS record type has been created and a new domain is defined to support these lookups (Walls, 2006). Figure 4.20 shows an overview of this general process.

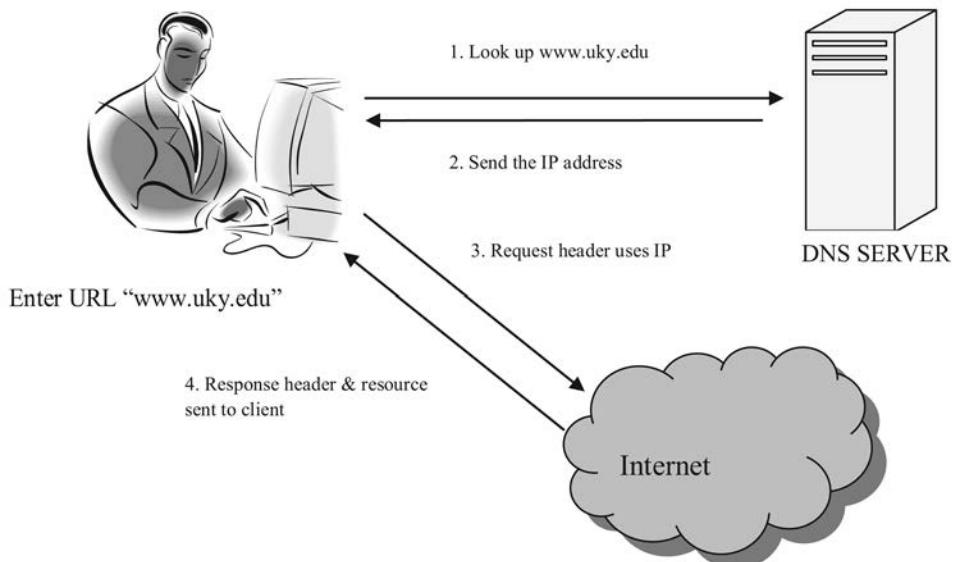


Figure 4.20 A simplified DNS lookup.

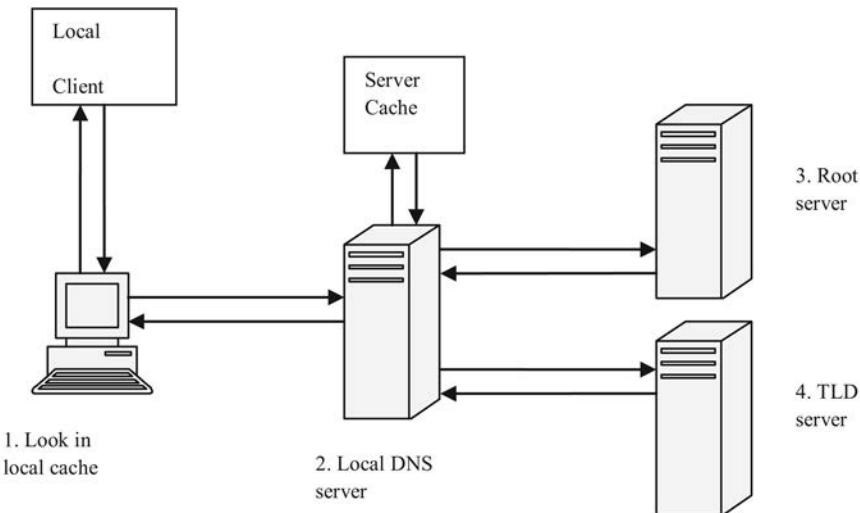


Figure 4.21 DNS lookups take place in a hierarchical fashion starting with the local cache, then to the local DNS server and cache, then to the root or other TLD server.

A demand for DNS lookups could become a significant bottleneck for Internet activities so both caching and a tiered approach are employed to make the process more efficient. The simplified lookup described previously actually takes place by querying a cache or a database at multiple levels as shown in Figure 4.21. In each instance, the request is passed up the chain until the needed lookup is found.

TCP/IP, SYSTEM LAYERS, AND OSI

Layer:
User
Application
TCP/IP
PPP
OS
BIOS
Hardware

Figure 4.22 System layers on an Internet-connected PC.

TCP/IP uses system layers according to the functionalities outlined in the OSI Network Reference Model. Figure 4.22 is a view of the system layers for a computer connecting to the Internet with PPP. When an Internet client application runs, it plugs into TCP/IP via a port assignment; TCP/IP plugs into PPP, which plugs into the OS via a socket API; and the OS uses its drivers to connect to the BIOS and the hardware.

The expanded TCP/IP layers are mapped to the OSI model as shown in Figure 4.23. In the implementation of TCP/IP, some OSI model layers are combined, resulting in four layers instead of the seven of OSI. This is not surprising because the OSI model is just a reference model that is adapted to many specific network implementations. The TCP/IP suite can be further expanded to include the common higher-level protocols as shown in Figure 4.24, some of which are discussed further in Chapter 5.

OSI Layer	Function	TCP/IP Layers
7. Application layer	What users see	4. Application
6. Presentation layer	Translation formats	
5. Session layer	Opening and closing of sessions	
4. Transport layer	Delivery of packets	3. Host-to-host
3. Network layer	Packets from data layer get addressed and routed	2. Internet layer
2. Data link layer	Packet formation and error correction	1. Network layer
1. Physical layer	Hardware and movement of bits	

Figure 4.23 OSI layers mapped to TCP/IP.

TCP/IP Layer	Protocols Associated with Layer				
Application	SNMP	SMTP	NNTP	TFTP	FTP
	HTTP	Telnet	BOOTP	DHCP	DNS
Host-to-Host	TCP		UDP		
Internet	IGMP		IP	ARP	
	ICMP			RARP	
Network	Just sees bits				

Figure 4.24 Overview of TCP/IP layers and common protocols.

COMMAND UTILITIES FOR TCP/IP

Ping is a utility run from a command prompt on Internet-connected hosts. A ping forms a message based on an ICMP echo request directed to a host so see if it is “alive.” If the host is available and not prevented from responding by a firewall, it bounces the message back (hence the name; *ping* is also a sonar term based on the reflection of sound waves by an object in water). The loop-back ping address (127.0.0.1) can be used to test the TCP/IP status of your local host computer. Another frequently used command is **ipconfig** along with the “**all**” modifier (written either as “/all” or “-all”) used in Windows. There are graphical options to reveal the IP configuration in the control panel, but the command version gives all the essential information in one screen of output. Another command is the **tracert** (or **traceroute** depending on the OS) command showing the path test packets take to a destination along with the number of “hops” (routers) needed. The **netstat** command gives the status of the ports in use. The **arp** command with the “-a” modifier reveals the physical addresses stored in the ARP table.

SUMMARY

The foundation of the Internet is TCP/IP. As of this writing, IPv6 is gaining acceptance but IPv4 is still in common use. Although there are key differences between the two versions of IP in use today, conceptually they both employ many common strategies and protocols. A basic knowledge of these technologies is essential to understanding how all the higher-level protocols work and how many information services are delivered via the Internet. TCP/IP depends on the key technologies of packet switching, IP addressing schemes, the use of subnet masks and subnet administration, and the DNS system. Private networks, VPNs, and proxy servers can extend IP address spaces or serve as an intermediary between a user and the Internet. Various command utilities are available to assist with configuring and troubleshooting TCP/IP Internet connections. All these foundations are referenced in the next chapter on the higher-order protocols associated with the TCP/IP suite.

REFERENCES

- Arnold, B. (2008). Sizing the web: Domains, sites, hosts. Retrieved May 1, 2008, from <http://www.caslon.com.au/metricsguide1.htm#domains>.
- Coffeen, Tom. (2013, April 4). A brief snapshot of IPv6 adoption. Retrieved June 7, 2013, from <http://www.infoblox.com/community/blog/brief-snapshot-ipv6-adoption>.
- Comer, D. E., & Stephens, D. L. (1991a). *Internetworking with TCP/IP Volume I: Principles, protocols, and architecture* (2nd ed.). Englewood Cliffs, NJ: Prentice Hall.
- Comer, D. E., & Stevens, D. L. (1991b). *Internetworking with TCP/IP Vol II: Design, implementation, and internals*. Englewood Cliffs, NJ: Prentice Hall.
- Expressing IPv6 addresses. (2005, January 21). Retrieved June 10, 2013, from <http://technet.microsoft.com/en-us/library/cc784831%28v=ws.10%29.aspx>.
- Fall, Kevin R., & Stevens, W. Richard. (2012). *TCP/IP illustrated* (2nd ed.). Upper Saddle River, NJ: Addison-Wesley.
- Fiocco, Alain. (2013, April). IPv6 adoption by the numbers. Retrieved June 7, 2013, from <http://www.slideshare.net/getyourbuildon/naipv6-summit-2013t-ipv6-adoption-by-the-numbers>.
- Kiernan, V. (2005). Missing the boat, or penny-wise caution? *The Chronicle of Higher Education*, 51(27), A33–A35.
- Kleinrock, L. (1996, August 27). The birth of the Internet. Retrieved October 1, 2007, from <http://www.lk.cs.ucla.edu/LK/Inet/birth.html>.
- Kozierok, Charles M. (2005, September 30). The TCP/IP guide. Retrieved June 11, 2013, from http://www.tcpipguide.com/free/t_IPv6DatagramOverviewandGeneralStructure.htm.
- Krol, E. (1994). *The whole Internet: Users guide and catalog*. Sebastopol, CA: O'Reilly and Associates.
- Leiner, B. M., Cerf, V. G., Clark, D. D., Kahn, R. E., Kleinrock, L., Lynch, D. C., et al. (1997). The past and future history of the Internet. *Communications of the ACM*, 40(2), 102–109.
- Loshin, P. (1999). *IPv6 clearly explained*. San Francisco, CA: Academic Press.
- Simoneau, P. (1997). *Hands-on TCP/IP*. New York: McGraw-Hill.
- Walls, Colin. (2006). *Embedded software: the works*. Boston, MA: Elsevier.

94 Internet Technologies and Information Services

- Warren, Christina. (2011, June 20). 9 Things you need to know about ICANN's new top level domains. Retrieved August 15, 2013, from <http://mashable.com/2011/06/20/new-gtld-faq/>
- White, M. (2000, February 2). Loans.com latest web name to make millionaire of seller. *Herald-Leader*, p. C2.
- Zakon, R. (2011). Hobbes' Internet timeline. Retrieved April 11, 2013, from <http://www.zakon.org/robert/internet/timeline/>

ADDITIONAL READING

- Hofstetter, F. (2005). *Internet technologies at work*. Burr Ridge, IL: McGraw-Hill.
- Leiden, C., & Wilensky, M. (2000). *TCP/IP for dummies* (4th ed.). Foster City, CA: IDG Books Worldwide Inc.
- Molyneux, R. E. (2003). *The Internet under the hood: An introduction to network technologies for information professionals*. Westport, CT: Libraries Unlimited.
- Young, M. L. (2002). *Internet: The complete reference* (2nd ed.). Berkeley, CA: McGraw-Hill Osborne.

Higher-Level Internet Protocols: Making the Internet Work

The higher-level protocols defined within the TCP/IP standard enable users to do activities such as email, chat, FTP file transfer, Web browsing, or VoIP. These higher-level protocols support the utility of the Internet and allow access to services and content, presenting a new array of acronyms to define, such as SMTP, FTP, and HTTP. The higher-order Internet protocols depend on the packet technology associated with TCP/IP, and they rely on client-server applications designed to utilize them; a good overview of these advanced Internet protocols is by Black (1999).

As discussed in earlier chapters, client-server architecture utilizes two pieces of software working in tandem for each protocol: the client software designed to formulate a request to the corresponding server, and the server software designed to reply to the request. All of this client-server activity happens through the formation and movement of packets (or datagrams) destined for some Internet host, and all are created and routed according to the rules of TCP/IP described in Chapter 4. However, there might be a number of client applications running on a host, and getting the packets to the correct application is the function of the port assignments in TCP, as discussed in the previous chapter. In that discussion, IP was introduced using the analogy of how the U.S. mail system moves physical packets using a standardized addressing scheme. The analogy was extended to TCP by considering how mail delivered to an apartment building needs additional information to ensure delivery to the correct apartment. This is analogous to port assignments associated with higher-level protocols for Internet packets; TCP ensures packets get to the right client or server application. Figure 5.1 shows the standard default port assignments for the FTP (20), SFTP (21), Telnet (22) and SSL Telnet (23), HTTP (80), and HTTPS (443) protocols. Each TCP packet is encapsulated into an IP packet for delivery across the Internet.

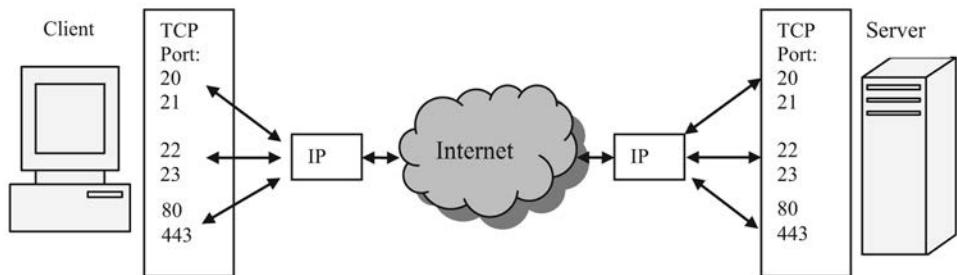


Figure 5.1 Common client–server TCP port assignments.

EMAIL: SMTP, POP, IMAP

Email has been, and continues to be, one of the most popular Internet applications. There are more than 2.1 billion email users worldwide, including 92 percent of American adults (Bright, 2012; Purcell, 2011). The Simple Mail Transfer Protocol (SMTP) makes Internet email possible. Email has been a popular application since the days of BITNET, which was a precursor to the Internet at many institutions. In fact, the incredible success of email has earned it the title of the original “killer app” of the Internet. It is somewhat ironic that something that was so high tech not long ago is now relatively passé to many of the millennial generation, who complain it is too slow and not as instantaneous as cell phone texting, computer IM, or other social media forums. Even with the growing use of text messaging and the ongoing spam problem, overall email use is still expected to grow to 2.7 billion users in the next four years (Buckley & Radicati, 2012).

Unfortunately, much email is spam, the generic term for unwanted and unsolicited mail. A little spam trivia: the term is derived from the classic Monty Python restaurant skit where all the breakfast options included varying amounts of the processed meat known as SPAM™, whether the customer wanted it or not. Spam is an ever-increasing and costly problem. A 2011 Symantec report estimated almost 80 percent of the email sent that year fit this category, which is not that different than a 2006 estimate of 75 percent of all email (MessageLabs, 2011). Spam is difficult to control or regulate because so much of it is from foreign sources; China and Russia are frequently identified as the country of origin for the majority of U.S. spam (Talbot, 2008). However, efforts to block spam at the server and client level have been moderately successful, as have occasional legal actions against spammers. One example of such intervention is the effort Microsoft coordinated against the Rustock botnet in 2011; shutting it down had the immediate effect of reducing total spam by about one-third at that time (Bright, 2011). These malicious spam-generating bots will be discussed in Chapter 6.

Early access to electronic mail was often through a shell account on a mainframe or other large-scale computer, which was not a very user-friendly environment. All ISPs now offer email service, and it is available through popular and free Web-based services such as Hotmail, which has transitioned to

Microsoft's newer Outlook.com, Yahoo mail, and Google's Gmail. The availability of email and cloud data storage that can be used from public access computers in libraries has helped close the digital divide for those without other access options. The alternative to Web-based mail is to use a specialized mail client such as Microsoft Outlook or other dedicated mail client program. These programs generally have more features for improved filtering and handling of email compared to Web browser mail systems, but they require separate software configurations for each user.

Mail servers handle the large traffic associated with email systems by using one or more protocols to implement SMTP. Two technologies for mail services are Post Office Protocol (POP) and Internet Message Access Pool (IMAP) server approaches. Another mail solution is the Microsoft Exchange Server option, which is a proprietary Microsoft enterprise solution that supports shared calendars and other tools. All allow users to view their mail using a mail client or through a Web browser interface; user preference determines whether the mail remains on the server account or is downloaded to a local computer.

SMTP was designed to transmit ASCII text data, and encoding schemes are needed to allow for the attachment of binary files. Examples of such schemes include Multipurpose Internet Mail Extensions (MIME) and UUENCODE. Both schemes convert a binary file into an ASCII representation for transmission and then convert it back to binary on the receiving end. This feature is both "good news and bad news" as binary attachments not only allow users to exchange data in all formats but also allow easy dissemination of viruses, Worms, Trojans, and other malware, which are explored in Chapter 6. These encoding actions require that both sender and receiver have the encoding technology as part of their mail program. Styled mail with much more formatting than simple text is possible by the insertion of HTML markup into the message to enhance presentation in the mail client. Because HTML is all just ASCII text, this approach works fine within the text-only constraints of SMTP as long as the recipient email program can render HTML.

The POP is now in version 3. It was developed to be an easy, quick way to download email to a personal computer. The earlier POP2, which goes back to the 1980s, required an SMTP server to send and receive messages; the POP3 can work with or without SMTP. It is not obsolete, but it was designed for an Internet that was used differently than today—one where users were usually using a desktop computer for their access. An email client such as the Eudora software was used to download messages to your computer, or with the later version, view it on the server without downloading. The good news was that downloaded email could be viewed offline, but the bad news was that downloaded mail was not available to view from other devices or locations. IMAP addressed this issue by defaulting to leaving email on the server, making it available anytime from any Internet-connected device. Email represents an essential communication channel for students, teachers, businesses, and the public.

Librarians were early adopters of email technology and found it applicable to many work situations. Email provides options for reference service, interlibrary loan requests, and professional development. Electronic discussion lists were an early application of email that allowed subscribers to join a community of people interested in sharing ideas, questions, and answers with

each other. They are still widely used, but blogs and social networking sites are more popular now and are displacing these email applications. Discussion lists may be moderated and strictly controlled, or very open and unregulated. Most discussion lists are run by programs such as the LISTSERV or Major-Domo at some hosting site. Because a program is managing the subscription process and options, list participants must learn how the program expects to receive instructions for list management activities; this is done either through email commands or through a Web interface. Those new to lists need to learn the culture of the list and the proper “netiquette” for such email communication in the context of established boundaries of acceptable activities for the list in use. Professional discussion lists expect participants to behave in a professional manner. This includes considering how the message is styled (such as not writing notes in all capital letters), the tone of the content (“flaming” other members of the group is not considered professional behavior), and understanding the list reply settings to avoid replying to all when intending to reach an individual.

Usenet groups still exist and at one time were a popular, albeit primitive, form of a virtual community made possible by the availability of email. Groups developed around almost every conceivable topic, and instead of having messages “pushed” to each member as in a discussion list, users posted messages to a common area where others could read them and respond. These groups are similar to the older “computer bulletin boards” that were quite common in the pre-Web era. The interest in Usenet groups has declined as other options such as blogs and social networking sites have become more common, but Google groups (groups.google.com) provides a searchable archive of more than 700 million Usenet postings spanning more than 20 years.

The blog, or the personal Web log, may feel somewhat familiar to those who participated in Usenet, but blogs are reaching a much wider audience; Pew estimated in 2006 that 1 in 10 U.S. adult Internet users had a blog (Lenhart & Fox, 2006). Interestingly, although blogging interest has declined among teens and Millennials it has grown to include 14 percent of U.S. adults by 2010 (Zickuhr, 2010). Blogs are no longer simply a personal journal for many individuals but a media source that has become increasingly important to information professionals. Blogs are made possible by the XML-based RSS protocol, variously defined as Really Simple Syndication, also Rich Site Summary, or RDF Site Summary. Discussion lists, Usenet archives, and now the blogosphere all reflect significant content channels available on the Internet; these content sources and Web 2.0 technologies are discussed further in Chapters 13 and 16.

REAL-TIME INTERNET CONNECTIONS

Several Internet protocols accommodate “real-time” connections enabling a “conversation” between client and server. The earliest is the telnet protocol allowing remote users to send commands to a host and see the command output. Although many today have never used telnet, they may use newer “real time” activities such as voice over IP and streaming that are made possible by broadband connections.

Telnet and TN3270

The telnet protocol facilitates a real-time connection between two Internet hosts. This was an essential tool needed to meet one of the early goals of the Internet, namely, to connect remote researchers to some powerful host such as a NSF supercomputer. The telnet protocol was important to libraries as a way to provide access to OPACs (Online Public Access Catalogs) in the pre-Web Internet, and many libraries provided this access. Libraries also used telnet connections with the Library of Congress catalog or the Online Computer Library Center (OCLC) as a way to import cataloging records for local use and to connect to fee-based services such as Dialog. These library telnet servers are no longer available and have been replaced with Web services. However, the telnet protocol is still used to perform other tasks on Web hosts such as resetting passwords or modifying file permissions.

There are several versions of telnet used for connections to the different types of Internet host, most commonly these were either IBM mainframe computers or UNIX systems. IBM mainframes were screen-mode machines, that is, they painted an entire screen of output at a time. UNIX hosts are line mode, showing output one line at a time. These differences in real-time output required separate versions of telnet; TN3270 for connections to IBM mainframes (3270 refers to a terminal emulation type) and standard telnet for line-mode UNIX systems. In addition, “secure shell” telnet that uses a different TCP port is available to provide the enhanced security required by many hosts for such a connection.

Telnet takes care of the connection to the remote host, but once connected, the user is required to login to an account on the host. Some connections to publicly available resources did not require a login or accepted a generic one such as an email address; this was useful for access to public services such as to early online library catalogs. In either case, once connected to the remote host, you had to know enough about the remote operating environment to make effective use of it.

RTP and IRC

The increase in broadband connections has stimulated interest in Internet-based delivery of live video feeds, video and music recordings, IM, podcasts, and VoIP. Initially, multimedia delivered via the Web required the client to download the resource fully before playback could begin. This combined with the limited bandwidth of dial-up connections made delivery of this type of content problematic. Multimedia delivery now uses *streaming* technologies. A connection is made to a streaming server, which then assesses the available bandwidth and calculates how much data have to be “buffered” on the client. This ensures that once playback begins, the incoming data stream will keep up with the media player output. The Internet protocol used for streaming is the transport level Real-Time Transport Protocol (RTP) and it utilizes User Datagram Protocol (UDP) in place of TCP. UDP has less packet overhead and prioritizes speed of delivery over error checking; streaming does not require reliable delivery because by the time packets could be re-sent, the player is

100 Internet Technologies and Information Services

usually past their location in the stream. There are additional standards working in conjunction with RTP, such as the Real-Time Streaming Protocol (RTSP), a standard for controlling streaming data on the Web, and Session Initialization Protocol (SIP) for VoIP. Internet Relay Chat (IRC) is another form of instant communication over a client–server Internet connection. Unlike the RTP protocol, IRC uses TCP packets. IRC enables a communication channel over which an online conversation can take place.

All these technologies had broad applications in libraries in their time. Although telnet as a way to access public information has been replaced by Web access, it is still used to interact with servers in a command mode to modify UNIX file permissions, run scripts, manage web servers, or change a password. Newer technologies such as IM and IRC are an option for reference and other real-time help, and podcasts and audio/video streaming are being used to support library instruction and tours.

VoIP is also a real-time connection; IP-based telephone now competes with the traditional Telco voice services. Some services simply enable a TCP/IP connection, which allows one computer host to “call” another, as with the Skype service. Many ISPs offer VoIP, providing the same service experience of standard analog telephone with the added benefit of eliminating long-distance fees. Many companies and individuals now elect to combine both their voice and data services using TCP/IP technologies.

FILE MANAGEMENT WITH FTP

The FTP allows the uploading or downloading of files between a FTP client and server. FTP servers accessed with FTP clients provided a way to access and share updated software drivers, programs, and data in the pre-Web environment. Files available on various FTP servers were accessed by using a known URL or by searching the Archie index discussed in Chapter 15. Early FTP sessions were launched from a shell account, and the user needed to know how to work in a command environment. Success depended on knowing where the file of interest was on the host, how to use the FTP client to issue GET or PUT commands, and whether the requested file was ASCII or binary. Modern graphical FTP clients make this process much more intuitive with a “drag and drop” interface that automatically detects the file type. Another feature of most FTP clients is an option for the manipulation of UNIX file permissions of uploaded files. As this protocol is frequently used to upload content to a Web server, being able to view and modify permissions is often useful. There are many free FTP clients available for downloading such as Cyberduck or Filezilla.

PROTOCOLS FOR INFORMATION SERVICES

The Internet has always been both a communication and information retrieval (IR) system. Early IR depended on file archives accessed with FTP or making a telnet connection to search a database. However, these early protocols did not attract many users. The evolution of the Internet to be viewed

primarily as an information delivery system that supports information seeking through browsing and searching was due to the development of the gopher and HTTP protocols. The Web protocols are what we depend on for these activities today, but gopher was an interesting precursor to the Web.

The Gopher Protocol

Gopher began as a local campus-wide information system developed at the University of Minnesota in the early 1990s. It used a system of hierarchically structured menus linked to resources. The development team was led by Mark McCahill, and as with the emerging Web, the developers of this service almost immediately recognized its applicability to all types of Internet resources and locations. Gopher was released in 1991 and it became an immediate success as many libraries, universities, and government agencies created Gopher services to provide Internet access to their documents. Yet, by the mid-to-late 1990s, Gopher had all but disappeared, supplanted by the Web. Obviously, the emergence of the Web had a huge influence on the decline of Gopher services, and its demise was helped along by the University of Minnesota's 1993 decision to charge licensing fees. In addition, the Gopher client software could not view the newly emerging HTML Web content, but Web browsers could view existing Gopher resources, adding to the obsolescence of the Gopher client.

During its short heyday, Gopher servers were setup at most institutions of higher learning and many companies. Menu systems were created, and these were then registered with the “mother gopher” at Minnesota. Gopher client applications became freely available for most platforms, and users could browse the list of all the Gopher servers, select one of interest, and then browse the hierarchical menus at the site. As with the Web, the number of servers and resources grew quickly and made browsing inefficient. Steve Foster and Fred Barrie at the University of Nevada developed the searchable VERONICA index, which enabled keyword searching of Gopher menu terms (Vidmar & Anderson, 2002). Supposedly an acronym for the “very easy rodent-oriented net-wide index to computer archives,” it was a companion service to the already-present Archie index of FTP servers, so the choice of the name also was a humorous reference to the characters from the Archie and Jughead comics. The VERONICA index was a precursor to the search engines of today, but it had several limitations. First, there were only a few mirror sites of this index, and it was consequently difficult to access as demand grew; second, the intellectual access to the resources was limited to an index built using only terms from the menus themselves, not the full text of the resource. Both these services will be revisited briefly in Chapter 15 on Internet search. Gopher’s day in the sun was short lived, but it represented an important first attempt to organize Internet content into a more useable and user-friendly IR system.

The Hypertext Transfer Protocol

The hypertext transfer protocol, developed by Tim Berners-Lee, is another key technology for the Web; it is the protocol that controls the interaction between

102 Internet Technologies and Information Services

client and server for Web resources. It is available in two versions: HTTP 1.0, which is still widely used, and HTTP 1.1, which has a persistent connection feature. Version 1.1 has not fully displaced version 1.0 for several reasons, including backwards compatibility for some servers and browsers, mobile applications that do not support it, and issues with how these protocols work with some proxy servers (Roberson, 2010). A more detailed summary of other key differences between the versions of HTTP is by Krishnamurthy, Mogul, and Kristol (1999).

There are eight different methods defined for HTTP; these are GET, POST, HEAD, PUT, DELETE, TRACE, OPTIONS, and CONNECT, briefly defined in Figure 5.2. In a standard client–server transaction, the Web client formulates the request header sent to the Web server via TCP/IP. This request has information about the resource, the method of the request, and information about the browser version. The Web server responds with a formal message followed by the requested content; the formal message is a standard response header preceding the requested file content. This standardized “conversation” takes place between client and server. The three methods for HTTP examined in detail here are POST, where data are posted to a program; GET, where something is requested from the server; or HEAD, where only the relevant response headers from the server are sent and not the full content of the referenced object. The HEAD method retrieves information about a transaction but not the actual resource itself. This method could be used to determine if a document or program expected to handle data is even actually present on the server, or it could get metadata about a resource such as the MIME type or the date last modified without requesting the resource itself.

GET —Requests a representation of the specified resource. By far the most common method used on the Web today.
HEAD —Asks for the response identical to the one that would correspond to a GET request, but without the response body. This is useful for retrieving meta-information written in response headers, without having to transport the entire content.
POST —Submits user data (e.g., from an HTML form) to the identified resource. The data is included in the body of the request.
PUT —Uploads a representation of the specified resource.
DELETE —Deletes the specified resource (rarely implemented).
TRACE —Echoes back the received request, so that a client can see what intermediate servers are adding or changing in the request.
OPTIONS —Returns the HTTP methods that the server supports. This can be used to check the functionality of a Web server.
CONNECT —For use with a proxy that can change to become a SSL tunnel.

Figure 5.2 Summary of HTTP methods.

As with all Internet protocols, HTTP is based on the transmission of ASCII text, and all communication between client and server takes place as a stream of 8-bit characters. HTTP version 1.0 manages four discrete stages in the client–server interaction; these stages are (1) a connection is opened via the default or other specified port; (2) the client sends a request; (3) the server responds; and (4) the connection is closed.

Stage 1 begins as with all Internet client–server interactions: a connection is made via a specific port; the default assignment for the protocol is assumed or if it is different, it must be specified with the URL. HTTP uses port 80 as the default (note that HTTPS, discussed later, uses a different default port), but other ports can be designated in the server setup. If a nonstandard port assignment is used, it is appended to the end of the URL string with a colon followed by the port assignment. The second stage is the client request, which is a structured message sent to the server. The message sent from the client requests service by sending request headers identifying the method, information about the client's capabilities to receive different types of data, and in the case of forms, the data posted with the form. For example, assume someone is looking at a page called **students.htm** found at the URL **http://www.uky.edu/CommInfoStudies/SLIS/students.htm**. Assume that page has a link to another page called **lisso.htm** that the user follows. In response to that action, the client software formulates a request header to send to the server. Figure 5.3 shows such a hypothetical client request header sent to a server. Sites such as <http://www.myhttp.info/> will show the information contained in a request header sent from your client.

The request begins with the specification of the method followed by the name of the requested resource. In this case, the server is directed to locate the file **lisso.htm**. The “body” of the message begins with a list of the various file types

```
GET /students/lisso.htm HTTP/1.1
Accept: text/plain
.
.
Accept: */*
If Modified-Since: Mon, 13 Oct 2013 15:00 GMT
Referrer: http://www.uky.edu/CommInfoStudies/SLIS/students.htm
User-Agent: Mozilla/5.0
[a blank line, containing only CRLF]
```

Figure 5.3 A client request header as sent to a Web server using method GET.

this client can accept; the ***/*** designation means it will accept any file type—it then is up to the local client to activate helper applications if the browser itself cannot handle the format. The **if modified** information instructs the server to send the resource only if the date/time stamp on the requested file is newer than that of a previously requested version of the file the client has in its local cache. The **referrer** information gives the location of the page with the link that initiated the process, and it is how the server determines where the requested file is in relation to the referring page. In this case, the server is directed to the file sought by removing the **/students.htm** part of the referring URL and replacing it with the string **/students/lisso.htm**, thereby building the absolute reference needed to locate the requested page.

The third stage is the server response to the request, which begins with the response header containing information about the transaction followed by the actual data. Server responses include locating a file requested in a GET, sending an appropriate error message if necessary, or accessing a server-side script to post data for further processing if needed. If the requested object is really at the location specified, the server responds with header information followed by the actual HTML resource. The site <http://viewdns.info/httpheaders/> allows you to view the response header of remote domain URL. A sample HTTP response header from www.uky.edu is shown in Figure 5.4. The server response begins with the “HTTP 1.1 200 OK” and provides HTTP version status information as well as a numeric code (200) indicating a successful transaction. Status codes for normal responses are between 200 and 299; codes 400–599 are error codes such as the infamous “404—file not found” server response. The response header includes date/time and server name/version information, as well as metadata about the resource being requested, which includes the length of the requested file in bytes and the file type, in this example, **text/html**. A blank line with an ASCII CRLF (carriage return/line feed) separates the header from the stream of bytes making up the requested file content, which would then follow the header information shown. The fourth and final stage is to close the connection.

```
HTTP/1.1 200 OK
Date: Sun, 11 May 2014 16:43:25 GMT
Server: Apache/2.2.15 (Red Hat)
Last-Modified: Sun, 11 May 2014 16:30:11 GMT
ETag: "d000f817-a781-4f92258453906"
Accept-Ranges: bytes
Content-Length: 42881
Connection: close
Content-Type: text/html; charset=UTF-8
Set-Cookie: BIGipServerukhome=950737068.20480.0000; path=/
```

Figure 5.4 A sample HTTP response header from www.uky.edu.

HTTP and Statelessness

From this discussion, it is apparent that in this first version of HTTP everything that happens on the Web consists of these short communication events between client and server ending with the closing of the connection, *at which time the server forgets everything that has just transpired*. There is no record of the transaction kept by the server; this condition is responsible for the *statelessness* of the Web. There is just one transaction per connection, and multiple connections are needed for even a simple Web page with a graphic. If there are six image tags referencing six separate images in the **lisso.htm** page, there must be seven trips to the server for each separate transaction between client and server: one for the initial HTML file, followed by six others to get each separate image file. One analogy highlighting the problems inherent in this approach for transactions is to imagine how a telephone conversation might proceed using the same rules. I would call up a friend (make a connection), ask if someone is there (make request), the person answering could say yes (response), and then hang up (close connection). I would have to call back, and proceed with the next cycle of request and response. This analogy is somewhat flawed in that it is at least likely a friend will remember who just called; this is not the case with a machine. This lack of connection memory is a problem for any kind of transactional activity such as online shopping or interactive searching where maintaining or mimicking a stateful connection is necessary.

Several solutions exist to overcome or compensate for the problem of statelessness. One is the use of cookies, which are “client-side persistent information.” Why is it called a cookie? According to Netscape, where the technology was developed, the state object is called a cookie “for no compelling reason.” A *cookie* is a small chunk of textual information produced by a server and stored in a cookie file on the client computer; the cookie can contain information about the status of a transaction or a profile you have given to a site. On mobile devices, the client object is not exactly the same as the one used by desktop browsers but instead involves a token set by the app and stored locally with HTML5 that can be retrieved later (Perez, 2013).

A few basic rules relate to cookie use: first, it is your prerogative to enable cookies; you can set the browser to refuse them. However, doing so can be problematic because cookies are used extensively by many sites. Second, these are just text files, and they can be located, viewed, or deleted. Some browsers create a file called “cookie.txt,” but different browsers handle cookie information in different ways. However, the file contents are not very informative because it contains codes that are intended for the server program requesting it. Third, only the server that sets the cookie with your client can subsequently request it, so theoretically, other servers cannot extract information about a transaction you had with another. However, cookies can be exploited by malware to hijack sessions; they can also be abused with ad serving third-party cookies or web bug HTTP-set cookie requests, discussed in Chapter 6. You can view a cookie by entering a JavaScript command in the IE URL location bar as you visit a page; Figure 5.5 shows a cookie set when visiting Amazon.com.

Some users are suspicious of cookies and believe they present a privacy issue. Cookies are widely used safely by many servers for online transactions.

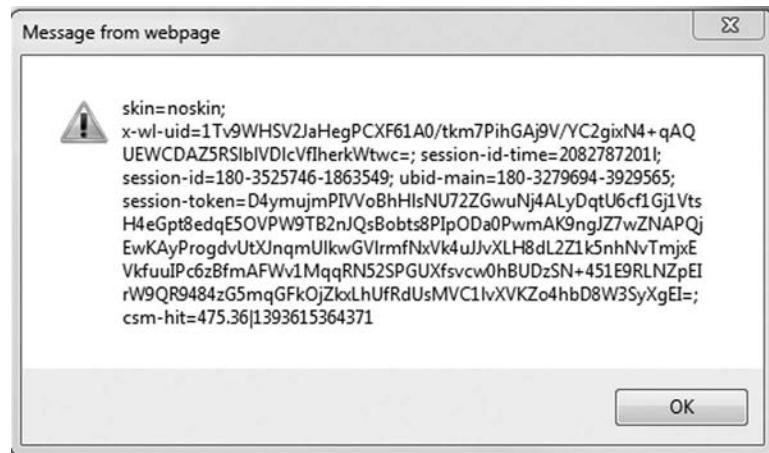


Figure 5.5 A view of the cookie set by Amazon when visiting that site, viewable by entering the line “javascript: alert (document.cookie);” in the URL location bar of IE.

However, they can provide a good deal of information about the shopper that retailers could potentially use for dynamic pricing (Klosowsk, 2013). In addition to cookies, hidden input tags in forms represent another method to pass transaction information back and forth between client and server. These tags are discussed in Chapter 8.

HTTP 1.1 specifically addresses the issue of statelessness by the addition of the “keep alive” feature, which enables client and server to maintain a TCP port connection for multiple transactions. Another advantage of HTTP 1.1 is it permits multiple requests to be processed simultaneously (a process known as *pipelining*). However, even with HTTP 1.1, the TCP connection can be lost or terminated for a variety of reasons, so cookies continue to be used in many Web transactions.

HTTP Secure

HTTP can take place via Secure Sockets Layer (SSL) or with Transport Layer Security (TLS), both of which provide encryption of data in network communications to prevent eavesdropping or tampering. Transport Layer Security is an IETF standard based on the earlier SSL approach developed by Netscape in the mid-1990s for its browser (Walls, 2006). Thus, HTTPS is not a separate protocol but instead are HTTP interactions that are combined with one of these standards. TLS and SSL involve the OSI session and presentation layers to establish a cipher and key for the session; these encrypted packets carried by the underlying transport layer. HTTPS sessions begin when a SSL enabled client contacts a SSL server, indicated with URLs beginning with **HTTPS** instead of **HTTP**. These URLs identify HTTP connections that take

place over TCP port 443 instead of the default port 80 and have higher packet security due to the addition of an authentication/encryption layer. The addition of the letter “s” to the standard “http” is sometimes overlooked when addresses are shared or linked, resulting in pages that either do not load or behave as expected.

SNMP and LDAP

Two other application layer protocols of interest to network managers are the *Simple Network Management Protocol* (SNMP) and the *Lightweight Directory Access Protocol* (LDAP), both of which are designed to gather information about the network or its users. SNMP automates the process of managing devices such as routers, computers, printers or servers on an IP network using agent software running on the device to store data in a database called the Management Information Base. When requested, the agent supplies that node-specific data to another program, the Network Management System that is run on a manager host (Mitchell, 2013a).

LDAP is another IP protocol that facilitates data exchange between programs to gather data from various network directories of users or devices. It is both a network protocol and architecture for organizing directory data (Mitchell, 2013b). For example, users on a network often need to find the contact information for other members of the network. Most dedicated email client software has a LDAP client to handle such queries to a contact directory service. For example, Microsoft uses its Active Directory (AD) service to authenticate users and store information about them and the network and the Outlook client uses LDAP to communicate with that directory database.

SUMMARY

The various client–server applications utilizing the higher-level protocols included within the TCP/IP family represent the programs people use for Internet communication or information seeking. Some of these protocols, such as Gopher and POP, are either no longer in use or are increasingly uncommon. Others represent specialized activities, such as uploading files with FTP, telnet sessions with a UNIX host, or configuring SNMP or LDAP servers, are of more interest to Web developers and network administrators than to average users. However, the SMTP, POP, IMAP, RSS, and HTTP or HTTPS protocols represent the backend technologies of the important activities of most Internet users. For most of us, client programs simply and transparently handle the protocols, making their role less obvious to the user. However, information professionals studying IT benefit from understanding the full range of all the Internet protocols that support the various programs used to perform these tasks. A deeper understanding of the HTTP protocol, and the related issue of statelessness in data communications, is critical to the delivery of transactional services. Web 2.0 technologies depend on adapting TCP/IP protocols and applications to the development of new services for users.

REFERENCES

- Black, U. (1999). *Advanced Internet technologies*. Upper Saddle River, NJ: Prentice Hall.
- Bright, Peter. (2011, March 29). Rustock repercussions: Spam down by a third, at least for now. Retrieved June 11, 2013, from <http://arstechnica.com/security/2011/03/rustock-repercussions-spam-down-by-a-third-at-least-for-now/>
- Buckley, Thomas, & Radicati, Sara. (2012, October 12). Email market, 2012–2016. Retrieved June 11, 2013, from <http://www.radicati.com/wp/wp-content/uploads/2012/10>Email-Market-2012-2016-Executive-Summary.pdf>.
- Klosowsk, Thorin. (2013, January 7). How Web sites vary prices based on your information (and what you can do about it). Retrieved September 26, 2013, from <http://lifehacker.com/5973689/how-web-sites-vary-prices-based-on-your-information-and-what-you-can-do-about-it>.
- Krishnamurthy, Balachander, Mogul, Jeffrey C., & Kristol, David M. (1999). Key differences between HTTP/1.0 and HTTP/1.1. Retrieved June 11, 2013, from <http://www8.org/w8-papers/5c-protocols/key/key.html>.
- Lenhart, A., & Fox, S. (2006, July 19). Bloggers: A portrait of the Internet's new storytellers. Retrieved October 1, 2007, from <http://www.pewinternet.org/pdfs/PIP%20Bloggers%20Report%20July%2019%202006.pdf>.
- MessageLabs March 2011 intelligence report. (2011, March). Retrieved June 11, 2013, from http://www.message-labs.com/mlireport/MLI_2011_03_March_Final-EN.pdf.
- Mitchell, Bradley. (2013a). SNMP. Retrieved June 12, 2013, from <http://compnetworking.about.com/od/networkprotocols/g/snmp-management-protocol.htm>.
- Mitchell, Bradley. (2013b). LDAP. Retrieved June 12, 2013, from <http://compnetworking.about.com/library/glossary/bldef-ldap.htm>.
- Perez, Sarah. (2013, February 25). Apple rejecting apps using cookie-tracking methods, signaling push to its own ad identifier technology is now underway. Retrieved September 26, 2013, from <http://techcrunch.com/2013/02/25/apple-rejecting-apps-using-cookie-tracking-methods-signaling-push-to-its-own-ad-identifier-technology-is-now-underway/>
- Purcell, Kristen. (2011). Search and email still top the list of most popular online activities Retrieved June 11, 2013, from <http://www.pewinternet.org/2011/08/09/search-and-email-still-top-the-list-of-most-popular-online-activities/>
- Roberson, Jarrod. (2010, January 15). Is HTTP/1.0 still in use? Retrieved June 10, 2013, from <http://stackoverflow.com/questions/2073392/is-http-1-0-still-in-use>.
- Talbot, D. (2008, May/June). Where spam is born. *Technology Review*, 111, 28.
- Vidmar, D., & Anderson, C. (2002). History of Internet search tools. In A. Kent & C. Hall (Eds.), *Encyclopedia of library and information science* (vol. 71, pp. 146–162). New York: Marcel Dekker, Inc.
- Walls, Colin. (2006). *Embedded software: The works*. Boston, MA: Elsevier.
- Zickuhr, Kathryn. (2010, December 16). Generations online in 2010. Retrieved June 11, 2013, from http://pewinternet.org/~media/Files/Reports/2010/PIP_Generations_and_Tech10_final.pdf.

ADDITIONAL READING

- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.

Internet-Connected Devices and Security

Computer and network security is a broad topic with many facets, and Internet connections add significant new security issues for computer and mobile device users. General security topics go beyond just dealing with malware. They include preventing intrusions into server and client computers, network eavesdropping and Wi-Fi spoofing, dealing with malware, authentication issues, maintaining secure Web transactions, protecting individual and organizational privacy, and securely administering servers. Cybersecurity affects not just computer users but all consumers, as evidenced by the data breaches of major department store systems and the Heartbleed bug that have potentially compromised the personal information or passwords of millions of customers. There are a number of related technical, ethical, and legal topics such as encryption and cryptography, respect for intellectual property, and the role of law enforcement agencies, especially in the context of expanding governmental initiatives and mandates. This overview explores only a few of these security topics, emphasizing various common threats, how Internet connections create new risks, and risk reduction strategies.

Internet connections create TCP/IP port issues that can be exploited in attacks and some specific examples of Internet-related threats are discussed. Security threats and solutions are often platform specific, and this chapter focuses mostly on Windows-based PCs and Android mobile platforms for the examples. This is not to imply there are not similar issues for Apple or UNIX systems, but it is primarily a reflection of the dominance of Wintel in the world market and the rapidly growing use of smartphones and tablets. Windows also is the platform that I have worked with the most in the context of these issues. Many workplaces, including libraries, have personnel dedicated to identifying and solving Internet security problems, but sometimes much of this responsibility falls to individuals who are not necessarily network or computer security

110 Internet Technologies and Information Services

experts. This overview is for the nonexpert who must become more knowledgeable about computer security issues.

Security has been a major concern throughout the history of computing, and computer viruses targeting the PC have been a problem for users from the beginning. While most people are familiar with the term *virus* in the context of computing, most threats are not true virus programs. *Malware* is a generic term used to describe any of the programs designed to annoy, damage, or invade your computer; these include viruses, worms, and Trojan horses as well as various types of spyware and adware. In addition to malware, social engineering threats such as phishing schemes and various forms of spoofing or pharming also pose security problems.

Transmission of early virus programs depended on users physically sharing infected disks. However, with the transformation of the personal computer from an isolated word processor and number cruncher into an Internet-connected communications device, many new avenues for transmission became possible. The networked and mobile Internet environment has created a more urgent interest in malware transmitted via networks, also known as Malicious Mobile Code (MMC). This broader term encompasses any program that can move from computer to computer via a network that is intentionally designed to modify a computer system without the user's consent (Grimes, 2001). There has been an explosion in various forms of malware; one estimate reported 5 million new variants were released in 2007, an increase of 4 million over 2006 (Heise Security, 2008). The Kaspersky Lab reports that this trend continued during 2011–12 (Cyberthreat forecast for 2012, 2011). Windows is a common target for these malicious programs, but malware is increasingly also affecting the Mac OS (Raiu & Emm, 2012). In 2012, it was estimated that about one in five Apple computers had malware, but most of it was intended for Windows machines; only about 2.7 percent of those Apple computers had Mac OS malware (Albanesius, 2012). As will be discussed later, Android devices are the more common target in the mobile environment. The emergence of the Web, the expansion of broadband connectivity and Wi-Fi hotspots, and the vast number of computers or mobile devices in use make it possible to share information and files around the world with the click of a mouse or a tap on a screen. Malware is often spread by simply visiting a malicious or compromised website. Symantec reports that such Web-based attacks increased by 30 percent in 2012; about 61 percent of sites that spread malware are regular sites that have been hacked or compromised, and that blog, personal, and small business sites tend to be compromised more often than large company or educational sites (Symantec Internet security threat report, 2011). It is this type of “drive by” attack that accounts for much malware dissemination.

While it is easier than ever to connect to any host on the Internet to share information or to set up your own Internet-based information services, it is also easier for *others* to find and connect to your device as well. Such intrusions may result in loss of personal data or system control. Many users have become painfully aware of this new reality with the arrival of a cheerful note from their network security officer informing them there are rogue servers running on their computer or that it has been turned into a spambot. It is important to understand how such attacks can happen and how to deal with them.

An increased awareness of the external threats to the networked PC is the first step for detecting and deterring them.

SECURITY ISSUES FOR PCS AND MOBILE DEVICES

The first widespread PC computer virus was the Elk Cloner virus written by a ninth grader in 1982. Given the somewhat flip response by some who suggest the solution to the well-publicized Windows vulnerabilities is to “get a Mac,” it is mildly ironic that this first virus targeted the Apple II. There are differing approaches and unique security issues for each of these platforms but both platforms can be attacked by malware. However, in general, Apple continues to be a more secure environment; the Mac OS does better at protecting system functions and uses a new version of OpenSSL for data encryption (Kerner, 2013). Apple’s market share has been growing in recent years, partly due to its popular iPhone, and new Apple converts may find they have fewer of the problems discussed in this chapter. The Windows examples given in this chapter reflect the current reality that this relatively ubiquitous platform is a more common target. Since that first primitive virus, there has been a long succession of new threats and corresponding responses, and the ongoing struggle continues between those who use programming to attack or annoy and those who produce programs to prevent or eradicate these problems.

The Windows and UNIX operating environment allows for multiple users with differing levels of authority to access the system. The most powerful access level is an account with administrative privileges. Administrators can set up other accounts, install programs and Internet services, and access, delete, or modify any file on the system. The TCP/IP standard is the foundation of the Internet, and it is part of the Windows OS. The Microsoft IIS (Internet Information Services), a suite of programs built into Windows, allows any desktop machine to function as a server that supports various Internet protocols such as telnet, file transfer (FTP), chat (IRC), and Web (HTTP).

Any OS has potential security gaps, and it may seem that Windows has a disproportionately large number of such security problems, a reputation that has not been helped by the discovery of a recent “zero day” vulnerability with IE versions 6 through 11 that could result in remote code execution unless a patch is installed (Security TechCenter, 2014). However, there are several mitigating factors to consider. First, it was an early priority of Windows developers to make advanced features easily accessible and this goal was sometimes at odds with security. Second, the huge worldwide base of Windows systems makes them attractive targets for hackers for practical, and sometimes philosophical, reasons. Because these powerful platforms are so ubiquitous and sometimes quite vulnerable, there are large numbers of hackers and malware writers who exclusively target Windows-based systems trying to exploit any weaknesses. In the mobile environment, Android seems to be the main target of malware; it is a very popular platform and it is relatively easy to create and distribute apps. Malware writers and hackers have at least one of many malicious goals, including data or identify theft, system vandalism, or rogue server operation.

TCP/IP AND PORTS

Many malicious programs or attacks utilize an available TCP/IP port. As discussed in earlier chapters, ports are an essential component of TCP/IP as all the Internet protocols depend on client–server interactions operating on a designated port. These Internet protocols use client–server architecture, and any given computer can run both client and server programs. The TCP/IP specification uses 16-bit numbers for ports; hence, there are over 65,000 ports theoretically possible. The port assignment is responsible for directing the separate streams of TCP/IP packets destined for the various client or server programs running a particular host. The common protocols have default port assignments, such as port 80 for HTTP; this assignment results in a Web server “listening” at port 80 for client requests. A process running in the background waiting to respond to an event is generically referred to as a *daemon*; a Web server listening at an assigned port is thus referred to as the HTTPd. The more secure version of the protocol is HTTPS, designated with “https” at the start of the URL string. As described in Chapter 5, HTTPS uses SSL or TLS to provide encrypted data communications via a unique TCP port. Many websites using the Apache and Nginx webserver software implement SSL and TLS with the open source OpenSSL. The Heartbleed bug exploited a vulnerability of the OpenSSL encryption process that could result in the capture of the passwords of users when they access affected servers (Strange, 2014). Although this is mainly a server security issue, the Heartbleed bug can also affect proxy servers, routers, and various client programs (Chien, 2014).

TCP/IP specifies a large number of unused communication ports that are potentially available for hackers and worm programs to exploit as a communication channel. In addition to TCP/IP ports, NETBIOS ports used by Windows network for file and print sharing functions can create security concerns. Even though many systems do not need these services, many Windows systems enable NETBIOS through TCP/IP. These ports can become a security issue by providing a conduit used by hackers and worm programs to extract information about your computer. The NETBIOS ports 137, 138, and 139 are sometimes referred to as “scanner bait” because hackers and worm programs often scan them to see if they are available for their intrusion attacks. One academic institution reported over 18,000 different sources were probing campus IP addresses on port 137 during a single day and that one IP address in Taiwan accounted for over 20,000 such attempts to access their campus computers in this way (CalTech Information Technology Services, 2003). Unfortunately, these activities are still common, and many technology departments block these ports at their firewalls. These ports have legitimate functions in Microsoft networks but they carry security vulnerabilities if they are enabled over TCP/IP on computers that do not need them for these uses.

The sequence often goes something like this. First, a vulnerable machine is located either via a virus attack, malicious scripts in a visited website, or with port scanning programs. A hacker may gain administrative access, by cracking or stealing passwords, by using programs to attack the system to upgrade privileges for another account, or by exploiting system or network vulnerabilities. A hacker could then steal data, monitor the user, or setup hidden server functions using the Microsoft IIS suite. These rogue server programs often are

assigned an unusual TCP/IP port, given legitimate sounding names (sometimes by renaming a legitimate Windows system file such as a Windows DLL), or hidden in some system directory where they are not easily noticed. The computer is now the base of operation for the remote user or is acting as a zombie machine generating spam or worse.

Attacks can also exploit the TCP protocol acknowledgement feature. As described in Chapter 4, TCP-based client–server communication involves a three-way handshake: the client sends a TCP SYN request, the server responds with a SYN-ACK message, and the lastly, the client acknowledges that server reply. This can be exploited in a denial of service type attack. A malicious server can be configured to send many SYN requests but not send the expected acknowledgement of the remote server reply. This leaves many “half open” connections that can overwhelm the server that is being targeted. Alternatively, the attack can involve the use of spoofed IP addresses in the IP packet that contains the SYN request; that host will not send the expected ACK message because it was not the true source of the original SYN message. Server administrators have a number of possible responses, including the use of firewalls and proxies (Eddy, 2006).

INTERNET CLIENT PROGRAMS AND SECURITY

All Internet activities depend on client programs tailored to specific activities. The list of these various client–server combinations is fairly long and covered in other parts of this text, but for most users the vast majority of daily Internet activities involve either a Web browser or an email client. Client-side program security options in these programs are extensive, but their success in large measure depends on the knowledge and subsequent choices of the person using them. The simple acts of surfing the Web and receiving email with attachments or links can be the start of various security problems. Websites can deliver pages with embedded scripts or applets resulting in spyware installations, and email messages contain phishing attempts and distribute malicious attachments. Many client-side technologies can be used for nefarious purposes such as embedded Java applets, JavaScript, Active X controls, and Visual Basic scripts. Technologies such as AJAX are used to create Web widgets, which is the term given to mini-applications popular in a variety of Web 2.0 tools. However, the use of AJAX technology is a source of concern because these widget applications can enable cross-site scripting attacks (Yaneza et al., 2008). Web browsers offer some protection by enforcing a “same-origin policy” that usually prevents a page at one domain from accessing information from another domain (W3C, 2010). However, there are exceptions to that policy; for example, the HTML iframe and script elements allow content from other origins to execute. Thus, the browser policy against such cross-domain activity would not protect data parsed within a script container used for JSON feeds (JSON is discussed in Chapter 12). Examples of malicious cross site scripting (XSS) are a modified URL sent via email that can steal cookie session data for a concurrent session when clicked on or a malicious script that steals cookie data to highjack a session.

However, Web developers frequently use these technologies for legitimate purposes to enhance the Web experience, so globally turning off client-side

script execution for all Web sessions is not a very practical option. Vigilance is called for, and some more specific and less draconian suggestions are offered in the following sections.

INTERNET SERVER SECURITY

This chapter is primarily on client-side security; Web server administration requires significant expertise with the platform in use because the consequences of a server security failure are very serious. An infected or compromised desktop computer misbehaving on a network can simply be unplugged from the network until the problem is resolved, but taking a server offline means lost productivity affecting many users. Many Web servers run on UNIX (or LINUX) machines, and the system administrator controls them. Security on these hosts begins with UNIX administration and there are many excellent sources of UNIX and LINUX (Kaplenk, 1999). Most of these server issues go well beyond the scope of this overview but a few general comments are included here.

The topics of port security, the implementation of server SSL and TLS, and the ACK/SYN feature of TCP are of particular relevance for anyone running an Internet server. As discussed earlier, the Heartbleed bug associated with a common version of OpenSSL affected millions of users of a huge number of websites including many popular social media sites such as Pinterest, Facebook, and Instagram (Mashable Team, 2014). Although an OpenSSL patch quickly was made available, server administrators also needed to revoke prior SSL certificates and regenerate new SSL keys (The heartbleed bug, 2014). Denial of Service (DoS) attacks are a common problem that often plagues server administrators. In addition, Web servers also might be performing other concurrent server functions. It is best to have dedicated computers for different major functions, so the administrator does not have to secure many different programs on the same server. Using strong password protection for server access is essential, and protecting access to the system “superuser” or “root” privilege account is critical. Default passwords are sometimes associated with new installations; these always should be reset to new highly secure ones. Services not requiring authentication by users such as anonymous FTP servers can be exploited by hackers and therefore should be setup by those who understand the risks and know how to minimize them. In addition, other server-side add-ons that are installed or enabled to run with the Web server may introduce security issues. For example, earlier versions of the PHP engine used for script processing had the “register_globals” setting set “on” by default, creating security issues. This setting default changed to “off” starting with version 4.2.0, and removed from PHP in the latest version to enhance the security of server script processing, but those changes created problems for older PHP scripts designed around the previous assumption. Another area of server-side security concern is the use of Common Gateway Interface (CGI) scripts because these server scripts can be accessed by, and accept parameters from, a URL submitted by a browser. Firewalls and VPNs are one strategy to protect servers from outside incursions. Many Web services require a high degree of transactional security to protect personal or financial data exchange between client and server, as highlighted

by some notable recent failures of companies to adequately protect customer information. Protecting transactional data usually involves encryption and authentication at the network or application level, and the Heartbleed bug demonstrated how that also can fail. Clearly, the topic of Web server security for large organizations is extensive and best left to the expert.

THREATS AND ISSUES

Computer threats and malware take many forms, from the classic virus to Flash cookies on websites. A few of the most common types of threat are described in this section.

Viruses, Worms, and Trojans

Computer viruses, worms, and Trojans are common forms of malware that can result in simple annoyance or a disaster that seriously compromises a system. There are many types of problem files afflicting the Internet-connected PC, all of which fall under the general label of malicious mobile code. The classic *virus* has executable code that acts as either a file infector or a boot sector infector. The virus code runs whenever the infected program runs or when an infected disk is shared. One common early transmission strategy was the boot sector virus. Windows disks have a boot sector outside of the data area that is not visible when the disk contents are viewed. A virus placed there was functionally invisible to the user, but when the disk was used on another computer, the virus was transferred to the hard drive boot sector and then copied to all subsequent disks used on the infected machine. Such an etiology is primitive by today's standards, but nonetheless, many PCs were infected in this manner. Although boot sector viruses are no longer usually spread by sharing disks, there are forms of malware that can utilize the boot record of a hard drive, called the Master Boot Record (MBR). For example, a type of rootkit can hide in the MBR and be loaded on boot up, placing itself between the OS and the hardware where it can then modify files as they are passed between the hard drive and RAM. By not modifying the file on disk, it becomes harder to detect (rootkits are discussed later in this chapter). Windows has built-in protections to prevent malicious use of the MBR, but rootkit developers continue to develop new techniques to try to circumvent those protections.

There are also *macro viruses* that can infect and spread via certain data files. These are typically written using the Visual Basic macro capabilities found in Office applications, thus making data files a possible source of a security threat. One of the first macro viruses was the Concept virus, named because it proved the “concept” that formerly harmless data files could spread a virus. The Concept virus infected the Microsoft Word template file **normal.dot** with the result that all new documents created and shared by the user contained the virus. The PDF file format can be exploited in a similar way—PDFs allow the embedding of many media types that may use JavaScript for rendering, execute files, and contain forms that can send data to a remote server, all which could potentially be abused by malware developers (Rubinking, 2010).

Worms and *Trojan horse* programs do not act in the same manner as classic viruses. They may cause any number of problems—they can turn computers into “zombie” spambots, send out thousands of replicated copies of themselves over a network, take part in “denial of service” attacks, or cause the afflicted host to lose data or crash (Trend Micro, 2008). The first widespread Internet worm program was the Morris worm, written by a Cornell graduate student in 1988. Since that time, there have been many other examples of Internet-based malicious mobile code. The Melissa virus of 1999 was one of the first to spread by email attachment; it would launch when opened and then send itself to the people listed in the user’s address book. In 2002, David Smith was convicted and imprisoned for writing and releasing this worm (Sophos Inc., 2002). The Love Bug, Code Red, Nimda, Blaster, and Sasser are other well-known examples of worms that were widely disseminated.

Worms use host systems to replicate and spread, often without the user being aware of this action, and the Morris worm quickly infected over 6,000 university and government computers on the Internet. Worms and Trojans are often lumped together, but they are somewhat different in action. As the name implies, Trojan horse programs are imposters; they claim to be something good but are, in fact, malicious. They depend on misleading the user to run them, and this misrepresentation is their defining characteristic. Unlike worms, they are nonreplicating, and unlike viruses, they do not infect other files. Nevertheless, if they succeed in deceiving the user, they can serve as an entry point for a future attack. The TROJ_PROXY malware program is an example; it propagates either as a zipped email attachment or through a visit to a Website with malicious scripting or applets. When run, it establishes itself as a startup service that connects to other malware sites for further programs or instructions. The result could be programs engaging in DoS attacks, initiating a spam blast, or stealing data (Yaneza et al., 2008).

Many worms such as KLEZ spread by an email attachment of an executable file that would infect program files and then use addresses found on the infected system to create “spoofed” email that were then sent out with the worm attached. Other examples such as Nimda, Sobig.F, Code Red, and Blaster were potentially quite dangerous and were often spread by email attachments, Web pages with malicious JavaScript code, and by port-scanning programs seeking computers with an available TCP 135 port (Chien, 2001). Sometimes these can work in sequence; for example, Nimda also attacked servers using holes left behind by a previous Trojan/CodeRed-II attack. These programs could create additional security openings by granting administrator rights to the “guest” account, giving a hacker full access to the system. Once such control was gained, rogue servers could be set up or data stolen. Many of these programs, such as the Swen worm, are downright sneaky; it arrived as an attachment to a message that appears to be a Microsoft Windows patch. This is highly suspicious—it is safe to assume Bill Gates does not have time to email you and Microsoft does not distribute patches in this way. Once started, the program began a phishing attack with a dialog box asking you to provide personal login and password information as shown in Figure 6.1; the program runs whether you respond with “yes” or “no” in the dialog box (Canavan, 2007).

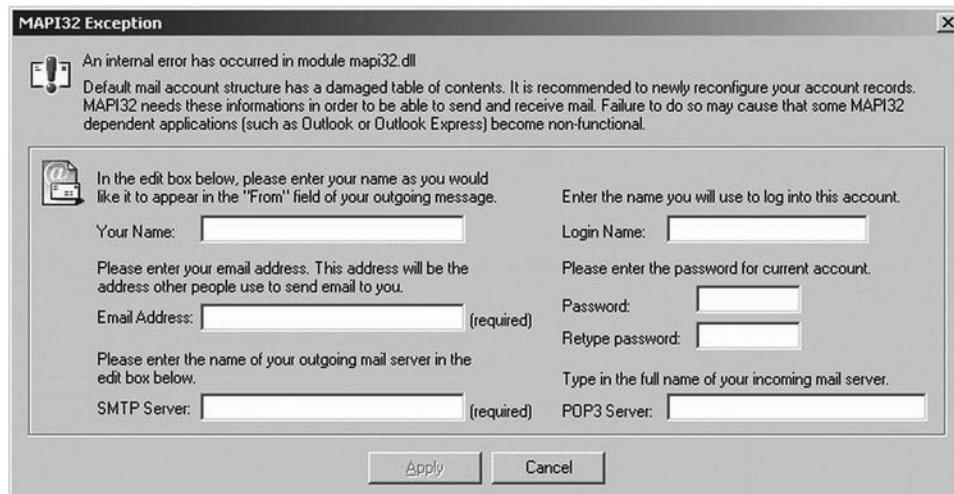


Figure 6.1 A screen shot of a spoofed Windows error message generated by the Swen worm.

Rootkits

Rootkits are Trojan variants that can highjack programs or gain control of a machine; examples include keyloggers and spambots. The “root” part of the name refers to how these programs can provide “root access” on a host. This type of rootkit is also known as a RAT (Remote Administration Tool) because of the potential backdoors it creates for root access. In Windows, rootkits can use system hooking to attach themselves to system files, processes, and registry keys to hide programs and activities from the user (Windows rootkit overview, 2006). They can vary in severity, but their removal is always a priority.

Rootkits can operate at either the kernel or user level. A common user mode technique is the modification of a Windows DLL file associated with the OS or a program. Kernel-level rootkits can be hidden very effectively because they can embed themselves in the OS itself instead of being run by it. Because of this, specialized software or manual removal techniques are often needed; many standard antivirus programs cannot remove them. Another type of rootkit is the hypervisor rootkit. The hypervisor was introduced as a VM manager in the section of Chapter 2 on virtualization. A *hypervisor rootkit* can turn this technology against the user by creating a malicious VM that can hide itself from the actual OS and control what the OS can do.

Spam and Phishing

Spam mail is a huge annoyance for email users; as noted previously, a Symantec report estimated almost 80 percent of the email sent in 2011 was spam (MessageLabs, 2011). Although still a huge issue, spam volume declined 29 percent in 2012 and was estimated at 69 percent of all email that year

118 Internet Technologies and Information Services

(Internet security report, 2013). Not all spam is a direct security threat, but some types of spam, such as phishing mail messages, are problematic. *Phishing* is a form of “social engineering,” which refers to any technique designed to trick individuals into revealing personal or other information. These are fake emails with address spoofing to trick users into believing the source is legitimate, enticing them to volunteer personal information. Most phishing is generically broadcast, such as the infamous “Nigerian email money offer,” offering to share a large sum of money in exchange for the victim’s bank account information, which is supposedly needed for the transaction. An example of this type of phishing offer is in Figure 6.2. However, more targeted forms of social engineering have been made easier by the vast amount of personal information people post on various social networks such as Facebook, and spammers appear to be moving their activities to social media.

Most users are aware of these well-known frauds, but even so, losses in billions have been attributed to phishing schemes. One survey found 3.3 percent of Internet users reported losing money to phishing in 2007 (McCall, 2007). Phishing emails may purport to be from PayPal, your local bank, or your organization’s helpdesk asking you to verify or update your account information. These can look quite authentic, using details such as a spoofed “From” address header that uses legitimate addresses, real logos or trademarks, and

Important read carefully

I am Mr. xxxxxx and I represent Mr. Mikhail Khordokovsky the former C.E.O of Yukos Oil Company in Russia. I have a very sensitive and confidential brief from this top (oligarch) to ask for your partnership in re-profiling funds US\$446 Million. I will give the details, but in summary, the funds are coming via Bank in Europe. This is a legitimate transaction.

You will be paid 20% as your commission/compensation for your active efforts and contribution to the success of this transaction. You can catch more of the story on this website below or you can watch more of CCN or BBC to get more news about my boss.

<http://www.mbktrial.com/>

<http://news.bbc.co.uk/1/hi/business/3213505.stm>

<http://www.themoscowtimes.com/stories/2005/04/11/041.html>

If you are interested, please do indicate by providing me with the Following.

Your Full Name:

Your Complete Address:

Date of Birth:

Confidential telephone number:

Fax number:

Your Occupation:

Sex:

I will provide further details and instructions.

Please keep this confidential, as we cannot afford more political problems.

Please do send me your response’s soon as possible via my personal email: xxxx look forward to hearing from you.

Regards,

xxxxx

Figure 6.2 An example of a phishing email.

appropriate URLs. However, hovering over the link in such a phishing message often reveals a completely different URL that the spoof address redirects to, often a numeric IP address pointing to a script. Figure 6.3 shows examples of these more devious types of phishing with notes claiming to be from the IRS (left) and a local helpdesk (right). Although phishing through email has declined from about 1 in every 300 emails to about 1 in every 400, the overall threat it presents remains high because as with spam, phishing is moving more into social networks, where it is becoming more targeted and sophisticated (Internet security report, 2013).

A reasonable response is to develop both a healthy suspicion of any email solicitation and to utilize the spam settings available on the mail server and/or the client software to block them. Many mail servers allow users to customize the spam settings at the server level in addition to the filtering and blocking capabilities of client-side software. Social network accounts and associated privacy settings must also be monitored. Aside from software settings, there are a few common sense practical behaviors to follow. Do not respond to suspicious unsolicited email messages; responding to spam with pleas to stop just results in more spam. The “auto-reply” mail option often used by people away from their office is a courtesy to those expecting a reply, but it can sometimes also automatically reply to spam mail. Do not open attachments or URLs accompanying suspicious messages; URLs in messages often have been shortened or spoofed, so always “think before you click.” Even messages from a trusted person can be suspicious—it is quite common to get an email or social media message that looks like it came from someone you know, but if the body of the message just has a link with a “check this out” note and no other message, it

<p>Tax Notification Internal Revenue Service (IRS) United States Department of the Treasury After the last annual calculations of your fiscal activity we have determined that you are eligible to receive a tax refund of \$184.80. Please submit the tax refund request and allow us 6-9 days in order to process it. A refund can be delayed for a variety of reasons. For example submitting invalid records or applying after the deadline. To access the form for your tax refund, click here. Regards, Internal Revenue Service Document Reference: (92054568).</p>	<p>CONFIRM YOUR EMAIL IDENTITY BELOW Sent: Wed 4/16/2008 5:13 PM To: customer@customeronlinehelpdesk.com</p> <p>Dear Edu Email Account Owner, This message is from edu messaging center to all edu email account owners. We are currently upgrading our data base and e-mail account center. We are deleting all unused edu email accounts to create more space for new accounts. To prevent your account from being closed, you will have to update it below so that we will know that it's a present used account.</p> <p>CONFIRM YOUR EMAIL IDENTITY BELOW Email Username : EMAIL Password : Date of Birth : Country or Territory : Warning!!! Account owner that refuses to update his or her account within Seven days of receiving this warning will lose his or her account permanently. Warning Code:VX2G99AAJ Thank you for using edu.</p>
--	---

Figure 6.3 Two examples of phishing emails that appear genuine seeking personal information

very likely is not from the person you think it is. Since spam is often the result of email address harvesters crawling the Web, many users try to hide their email address from these programs by inserting an image for the “@” sign; programs cannot utilize an address with an image. Browser software should be set to prevent, or at least warn, of any attempt to install add-ons; for IE, zone settings should be set as high as can be tolerated, at least to the “medium” setting, and third-party cookies should be refused.

The balance between higher security and convenience involves constant trade-offs. For instance, adjusting spam filters to their maximum levels can flag legitimate mail as spam, requiring added effort to identify and retrieve it. Browser security settings may cause interruptions with dialog boxes asking for explicit permission to perform some action or run an applet. Antivirus programs can interfere with legitimate software installations and need regular updating. For now, there are no perfect solutions.

Hoaxes

Further complicating our computing life, some threats turn out to be hoaxes. The JDBGMGR hoax is an example that was propagated by well-intended but uninformed users being tricked into believing they have sent you a virus after they received the hoax email. The cycle began with receipt of an email from an email address you recognize warning you that they had been infected by a virus and they might have unknowingly passed it to you. It claimed you should look for a specific file on your computer called JDBGMGR.EXE and that if you found it, you should delete it and warn others you may have sent it to them. However, the file in question was a legitimate Windows program (the Java debugger), and the warning you sent to others starts the hoax cycle anew (Sophos Inc., 2008).

Fake Sites, Pharming, and Honeypots

Fake sites are counterfeits that attempt to mimic a legitimate company or present the appearance of some official government site. *Pharming* is a companion technique that guides or redirects users to these fake sites by poisoning a DNS server, ARP table, or local hosts file with fraudulent addresses or by embedding the link in a spam message or phishing attempt. The goal is to fool users into providing personal or financial information such as credit card numbers or bank accounts to the counterfeit site. Never trust a URL in an unsolicited email to be genuine. Honeypots are servers that are designed to be easy prey for hackers; they are used by security experts to study how intruders attack systems and to gather forensic information that may lead to their prosecution (Even, 2000).

Cookies and Web Bugs

As discussed in Chapter 5, many interactions on the Web depend on the exchange of cookies between a client and server. Typically, cookies are set by the server you visit and can be retrieved only by that server or when a user makes a return visit. However, site advertisers can set ad serving, third party

cookies. These can then be retrieved when visiting some other site with that same ad. Ad-serving cookies can be used in this way to track previous websites you have visited and then use the information to generate additional targeted advertising. Although this is not necessarily harmful, many users consider such tracking intrusive. A related concern about cookies is that the information they carry about your location, shopping habits, or brand preferences can be used to facilitate “dynamic pricing” resulting in different prices for an item for different shoppers (Klosowsk, 2013).

A Flash cookie is similar to a traditional cookie but this local shared object is set and used by the Adobe Flash player. It is a concern because it is not handled by many cookie blockers and can collect data or even activate a webcam on a computer. Because there are documented cases of stalking or spying via a computer webcam on a hacked computer, some users go so far as to cover the webcam lens when it is not in use. A *Web bug* is a very small or transparent embedded graphic that is associated with an HTTP set-cookie request by a third-party server; this is often an ad server but could also be associated with hackers collecting information to be used in some potential future attack (Parsons & Oja, 2012).

Bots and Spyware

Generically, a bot refers to any program that can automate some task. If it appears to do so with some intelligence, it is sometimes called an agent, or a bot that acts on your behalf. However, when such a program acts under the control of some remote hacker, it is referred to as a zombie. An example of a “good” bot would be a program that can automate Web searching based on parameters you provide. An example of a “bad” bot is one used in a DoS attack or as a spambot. Bots are particularly troublesome when many malware-infected computers work together in a botnet. At its peak in 2011, the Rustock botnet was believed to be responsible for about 30 billion spam emails a day (Bright, 2011a). It is estimated that as much as 88 percent of all spam is sent by these botnets, and that only about 10 such botnets might account for 74 percent of that total (Bright, 2011b). Botnets can also steal data, such as the key logging Pony botnet that is thought to have compromised data from over 90,000 websites, capturing information from more than 2 million users of major sites such as Facebook, Google, and Twitter and sending it to a hacker-controlled server (Dragani, 2013).

Spyware is malware that uses one or more of these techniques to track your activities and gather personal information from your computer, usually to sell or use for targeted advertising. However, these programs can be potentially much more dangerous as they can capture keystrokes to gather passwords or credit card numbers. Spyware can enter a system with freeware downloads or through popup ads.

Wi-Fi Eavesdropping and Spoofing

Wi-Fi networks use radio waves to broadcast both the presence of the network and the data moving within it. As discussed in Chapter 3, Wi-Fi networks

broadcast the SSID name as well as whether it uses encryption. Wi-Fi networks should use a strong password and the best encryption option the router supports to prevent unwanted access and potential eavesdropping. However, public wireless networks may not always use encryption or passwords. Public network users are also susceptible to spoofing by an “evil twin” network. This is when a hacker sets up a wireless hotspot with a deceptively similar name within the same range hoping to fool users into connecting to it. Once connected, all traffic is going through the hacker router, potentially allowing them to capture your personal or financial information. Always confirm the actual network name and if encryption options are available, WPA2 is preferred, followed by WPA, and then by WEP. If you use public networks such as those in cafes or airports for important work, you should consider using a third-party VPN client to ensure higher security (Lynn, 2010).

Mobile Device Threats

Malware and its related security problems have quickly followed users to the mobile world with about 150 known threats that target these devices (Dilger, 2013). As mentioned earlier, the vast majority of these issues are associated with the Android system, mostly because Apple more tightly controls third-party developers. The Android system is appealing for malicious code writers because it is widely used, relatively easy to work with, and has multiple distribution sites. By contrast, the Apple iOS is a closed system that distributes apps from a single source. It is possible for malware writers to sneak undesirable code into the App Store, but Apple has tighter control of its app-distribution environment.

Android malware spreads via apps that act as Trojans and through spam email messages that are read on the phone. Some of these programs can intercept and forward text messages. Kaspersky Labs reports a powerful malware app called Backdoor.AndroidOS.Obad.a that is designed to exploit multiple weaknesses in the OS (Cyberthreat forecast, 2011). It probes the device looking for Internet and root access. If successful, it reportedly can send text messages, delete replies, download and install files, receive the phone account balance, capture personal data, or even allow a remote shell in which the hacker can execute commands (Whitwam, 2013). Symantec reports that 32 percent of all mobile threats steal data (Internet security report, 2013). Suggestions for protecting these devices include locking the screen, installing an antimalware program, not caching passwords, being extra diligent if you root the device (i.e., allowing apps to run with root access), and installing apps from trusted sources (Mills, 2013). Given the emerging dominance of mobile technologies, it is likely these types of threats will grow and users will need to be diligent just as PC users have needed to be.

ANATOMY OF AN ATTACK

There are two different, but interrelated, types of attack. The completely automated attack is from malware introduced through email or a website

visit. In this case, the damage is solely from the invading program. Sometimes these attacks are to steal data or in the case of ransomware, extort a payment to unlock the computer or files. However, this type of attack is frequently the precursor to a direct attack by hackers who want to use the computer for their own purposes. Such targeted attacks continue to be a growing threat, increasing by 42 percent in 2012 (Internet security report, 2013). One type of targeted attack is the “watering hole” attack that begins with profiling a victim to identify websites they are likely to visit and then compromising one of those sites by injecting script code or HTML to redirect the victim to another site with the damaging malware. The compromised site is thus like a “watering hole” where a predator waits for a victim (Internet security report, 2013).

Hackers can get control of a networked computer through several approaches involving local or remote techniques. Local issues relate primarily to the level of system access possessed by someone has on a computer designated for use in a public area. Administrators achieve protection of these computers by effectively “locking down” the computer with strong passwords and rights policy management. In addition, computers in public areas are often managed by ghosting programs that reconstitute the entire drive each day, or even multiple times each day, to ensure it retains preferred settings with a clean start. The guest settings influence the prospects for a remote attack, but there are TCP/IP-related vulnerabilities that can sometimes escape attention. Because of this, there could be many computers in physically secure environments such as offices or homes that users naively believe are not at significant risk. However, there is a possibility of a remote attack on such a machine, perhaps by someone halfway across the world. Such attacks no longer require a great deal of expertise because there are many easily available tools that can make even the novice hacker quite dangerous.

How can hackers identify a vulnerable machine? It is surprising how much information can be obtained from a networked computer to programs that just ask for it, often for legitimate reasons. The IP address and a list of the available TCP/IP ports are two critical types of information easily obtained. Once a machine has been identified as vulnerable, the next step is to gain administrative access either through guessing weak or nonexistent passwords, through the use of hacking programs such as CRACK (a password decoding program), or with other programs designed to exploit security holes. For instance, programs exist that can upgrade privileges of the built-in guest account to give it administrator rights or that can record passwords and send them to a remote user. Trojan horse programs, Internet worms, and other viruses can play a role in this process. An attack might begin with exposure to malware that both identifies a potential host and provides the entry point for a hacker or a rootkit program, described earlier. If a rootkit results in granting administrative access to a hacker, the remote users can do just about anything they want; one goal might be to set up one or more Internet services to support chat with IRC or FTP sites for music or other materials. Such rogue servers can operate in stealth mode for some time before detection and final shut down. The security breakdowns allowing this to happen usually relate to security failures in one of four critical areas: OS and application maintenance, antivirus program currency passwords, and TCP/IP port security.

SECURITY RESPONSES

Although surrounded by many threats, users need not be helpless victims. There are a number of best practices and specific actions that can help minimize these problems. One simple solution is not to leave your computer constantly on and connected to the Internet, but given that most devices are constantly connected, there are things we can do to protect our systems during those times.

Antivirus Programs

One main line of defense for individual users against all types of attack is up-to-date antivirus protection. However, because most antivirus software depends on file recognition based on a definition file, one is always at risk of attack by a new virus or a variant the antivirus program has not seen before. An alternate solution is to run added software that blocks any nonauthorized program from running. The downside of that approach is it can interfere with the running of legitimate programs that have not been added to the list of permitted software, which means you must know if it is wise to override the protection software whenever a program asks permission to run. Antivirus programs usually include antispyware and ad-blocking features, but there are also programs just for spyware and adware issues. Such software should be used for both real-time protection and periodic system scans.

For Windows, the Microsoft Malicious Software Removal Tool and Windows Defender are available from support.microsoft.com. Companies such as Symantec and McAfee are well-known vendors for security software, but there are free options as well, such as AVG or Malwarebytes. However, one note of caution—there are websites that distribute malware through a popup notification that your machine is infected and that you should download the offered software. Such unsolicited third-party popup warnings that occur when visiting a website that are not from an already installed antivirus program should generally not be believed and can be a warning sign of a malwaresite.

Firewalls, Proxies, Routers, and VPN

Firewalls are another front line defense that allow for packet filtering as well as controlling TCP/IP port access. Windows has a firewall feature and there are other third-party programs. Firewalls block unneeded ports, filter packets from suspicious hosts, and report any intrusion attempts (Parsons & Oja, 2012). In addition to activating the firewall feature, additional protection can be gained by turning off Windows sharing features if they are not needed.

As described in Chapter 4, proxy servers and routers add security by using NAT to present a public IP address to the outside Internet but using private addresses for in-network devices. This process effectively shields your computer address. The use of a VPN client adds encryption for added security.

OS and Office Application Updates

As many threats target Microsoft systems, it is imperative to take advantage of the service packs and patches to respond to security issues as they are identified. Updating the OS with the latest patches is essential and it is one of the easiest security measures to implement. Service Packs refer to major updates to the OS with a large number of bundled updates and fixes. There is a steady stream of specific patches to fix immediate problems. You should also update Microsoft applications as well; Outlook, IE, and Office have security issues that require updates. Users can go into the IE “Tools” menu option and select “Windows Update.” This connects to a Microsoft website that scans a system, identifies critical updates, and downloads them for installation. The best strategy is to automate this process. In the “Control Panel,” there is an option for “Automatic Updates” where one can schedule a periodic check of a system—once a week is reasonable unless there is a specific threat announcement. However, note that administrative access to the system is needed to schedule this or to do major OS updates. Microsoft never sends out broadcast email messages with patches attached; emails purportedly delivering patches are fakes and an attempt to distribute malicious code. Keeping up with OS security is similar to dealing with a leaky roof; if it is raining all the time, ongoing vigilance is required. Good maintenance practices require that not only must the damage be minimized when a problem is discovered, but that preventative patches are regularly applied to address potential problem areas before they are exploited.

Password Security

Authentication in the form of user IDs and passwords is an essential part of protecting systems and your information from outside threats. Authentication can take multiple forms—it can be something you know such as a password or PIN, something you possess such as a swipe card, or use something about you physically such as facial recognition, fingerprints, or a retinal pattern. Higher security is attained when more than one authentication protocol is in use; for example, Google offers a two-step authentication method that has been also adopted by Apple and Microsoft (Cheng, 2013). A one-time security code is sent to your cell phone for the account and you must verify your identity with that trusted device before you can login from another device or make account changes; this is a stronger solution than the common “security” questions often used to change account settings. However, for many users accounts are accessed simply with something they know. Passwords are commonplace solutions to authentication. There are alternatives to traditional passwords; for example, Windows 8 allows for a picture password where reproducing three saved gestures on an image allow entry.

To be effective, passwords should be unique for each system or device, strong, and changed regularly. However, these requirements make them hard to manage and to remember. Password management has become the bane of

computing, but nonetheless it is potentially the weakest link in the security arena and they should be created and managed with that in mind. Beyond the extreme examples of bad passwords such as the word “password” itself, even those that on the surface seem better are easily cracked. A recent list of the 25 most popular passwords includes other equally predictable passwords such as “123456,” “admin,” “qwerty,” and “letmein” (Morris, 2012). Analysis of the data stolen by the Pony botnet revealed that such poor password choices are still commonplace (Dragani, 2013). Any common word or phrase is subject to a simple dictionary attack, run by a program that simply and quickly tries all words in a dictionary. Social engineering also comes into play; for example, it is often easy to gather password guesses such as birthdays or pet names posted on a social networking site. User IDs and passwords that are sent unencrypted over a network can be collected by packet sniffing or by keylogger programs.

Creating, and then remembering, strong passwords is a challenge and a tiered approach is reasonable with high-strength passwords used for important accounts and lower strength ones for less important ones, such as to a free news site that simply requires registration. It is important to maintain different passwords for different systems so that the loss of a single password would not provide someone access to every account you have. Any password can be subjected to a brute force attack with password cracking software, which is both widely available and sophisticated. Cracking programs use rules and word lists to make them more efficient, but in general, the longer the password is, the longer it takes to crack; each added character exponentially increases the cracking time. The standard minimum length of eight characters can be cracked relatively quickly but a 12-character password is much harder to break, taking nearly six months to crack by brute force (Anderson, 2013). Suggestions for passwords include using a passphrase such as the first letter or two of a song lyric or quote, but stronger options would be a more random passphrase that is easy to remember but not easy for someone to guess. In addition, using a phrase that is a truthful statement about you or your life is also less secure than a made up one (Brodkin, 2013). If a common word is used as part of a password, try reversing it—for example “kroYweN” for “New York.” In addition, you can substitute some letters with numbers or special characters, such as a zero for the letter “O” or a \$ for “S.” Another suggestion is the “underwear rule”—that is, change passwords regularly, and do not store them where others can see them (Griffith, 2011).

There are sites that generate random strong passwords for you or that test the strength of those you use. Storing and tracking all our passwords should be done securely—something better than the post-it note on a monitor or a Word file called “passwords” on the desktop is needed. There are both commercial and open source or shareware programs that can help with password management and security. Password managers can generate strong passwords, store them securely in an encrypted file, and use a browser plugin to provide them to the site. The key piece is a single master password that controls access to the password manager; obviously, that master password should be strong and stored securely. Services such as Lastpass (<https://lastpass.com/>), Keepass (<http://keepass.info/>), or the hardware-based Yubikey (<http://www.yubico.com/>) are examples of well-reviewed options.

Another authentication issue faced by Web developers is the need to ensure that their site is being utilized by a person and not a program. CAPTCHA (Completely Automated Public Turing Test to Tell Computers and Humans Apart) tests are the obscured or distorted strings of text that users must read and type before proceeding. However, people often find these hard to read and they raise accessibility issues. There are alternatives to CAPTCHA that can defeat bots, such as a math puzzle, a trivia question, or a second confirmation step, but each strategy has its own problems and tradeoffs (Pogue, 2012).

WARNING SIGNS OF A PROBLEM

If a scan with your antivirus software detects a virus on your computer, the best strategy is to go to “red alert.” One must immediately begin dealing with the problem by doing whatever is required to remove the virus and correct any changes it made to the registry or other system components. Infection by Trojan horse or worm-type viruses can be a prelude to a hacker gaining full access to your computer and should not be taken lightly. A few general “red flags” might be:

- Unusual activity on your machine (but be sure you know what’s “unusual”). This includes the sudden disappearance of free disk space; the appearance of new user accounts or folders; or a sudden, dramatic change in how long it takes a system to start up or shut down.
- Services “listening” at unusual ports can indicate a problem, as can the appearance of new services such as FTP or IRC. Unusual packet volume associated with the system, if monitored, is also a red flag.

Rubenking (2013) lists seven common warning signs that relate to what might constitute “unusual” activity, including the appearance of popup ads even when no browser is open, browser navigation being redirected; popup “security” warnings from a program you have not installed, posts you did not write appearing on your social media pages, not being able to use common system tools, and a program holding your computer for ransom. Of the various warning signs, one important, but hard to interpret sign is unusual port activity. For most users, the easiest way to check out a system is to go to one of the various Web-based security-scanning sites such as the Gibson Research ShieldsUp! (www.grc.com). These sites scan your system and identify vulnerabilities such as high-risk ports accessible to the outside world. For those comfortable examining their systems in more detail and interpreting the results, the Windows **NETSTAT** command or some separate port-scanning program such as the Active Ports freeware program are useful tools for examining ports in use.

To use the **NETSTAT** command, go to a command prompt, enter “**netstat -ano**” and press enter. The “**-ano**” portion of this command identifies a set

of three specific command modifiers (to see all these modifiers and their use, enter “**netstat -?“**). These modifiers provide added information, including the Windows process identifier, called a PID. This command identifies all ports “listening” for services. Not all strange ports are indicators of problems; legitimate programs such as mail clients use some unusual port assignments. To see what program is using such a port, you can go to the Windows Task Manager and lookup the PID to reveal what program is associated with the port. Although NETBIOS ports can be exploited, just seeing they are in use is not always a sign of trouble. However, finding new services such as telnet, FTP, or IRC servers assigned to nonstandard ports is a strong indication that the computer has been compromised, and appropriate remedial action is called for.

If you suspect a problem, what should you do? For a machine you control, first, take the suspect system off the network so that it is not an immediate threat to others. Then confirm the exact nature of the problem. Many sites such as **snopes.com** help distinguish hoaxes from real threats and the government National Vulnerability Database site (<http://nvd.nist.gov/>) tracks many problems and has many resources. If a problem is confirmed, the remedy may just involve applying an appropriate security update or using a specific malicious software removal tool such as the Microsoft Malicious Software Removal Tool or another source such as Malwarebytes. You should also run a rootkit revealer program if one is available for your OS since rootkits are not always detectable by standard antivirus programs. A short-term solution that can sometimes buy time is to boot into a command view and use the system restore to go back to a previous configuration to allow you to run troubleshooting programs with Windows. You should also purge all stored passwords and establish new ones for all accounts as soon as possible. If the source of the problem can be identified, it should be reported to the hosting service or site, if it is a reputable one. If the system has actually been hacked and you have data backups, it may be best to wipe the system clean and start over. Such radical treatment may be called for because just applying patches or removal tools does not guarantee the elimination of the threat. The use of ghosting software alluded to earlier makes it easy to reconstitute the original state of a compromised computer. If it must be done manually, back up all data files to prevent data loss, reformat the system, and install a clean version of the OS. After reinstalling the OS, immediately update it with all appropriate service packs and patches. If it was absent, install antivirus protection and update it, and then install the added protection of a firewall at the machine level. However, with local firewalls, be aware some settings can cause unexpected problems for computers on organizational networks. For instance, the default firewall settings may not allow the host to respond to a **ping** request (these are test packets sent to a destination address to see if it responds). On some networks, periodically responding to a ping may be required for the computer to retain the IP address assigned to it, so denying this response could result in the loss of the IP assignment. It is best to check with network administrators before implementing a firewall restricting ICMP activities such as ping.

The networked world gives people almost unlimited access to information, but the access comes with a cost. One cost is the need for ongoing vigilance to protect the security of your personal and workplace computers proactively.

The old adage of “an ounce of prevention being worth a pound of cure” is especially true in the world of the networked PC.

SECURITY “TOP-TEN LIST”

The following top-ten list of easy-to-do activities reflects a good starting point for the security-conscious user.

1. Use secure passwords for administrative access to the computer and all Internet accounts of value. Do not use the same password for all accounts.
2. Implement the principle of “least privilege”—consider using tiers of differing password strength for different systems. Consider using a nonadministrator account for some activities on some devices. Disable the guest access to devices and networks if it is not needed.
3. Know your machine! Have an idea of how much disk space you have available, how many user accounts are on it, who has administrative access, as well as what programs and services are installed.
4. Keep the Windows OS up-to-date with the latest patches and fixes.
5. Use a password for your Wi-Fi network and enable the strongest encryption available. Be cautious when using public Wi-Fi; confirm the network name with an employee.
6. Use antivirus software, and keep it current.
7. Use a local firewall if it will not interfere with other network functions in your environment.
8. Investigate disabling NetBIOS ports over TCP/IP, and turn off the Microsoft file and print sharing features if they are not needed.
9. Learn how to check or monitor port activity with the “netstat” command, some port scanning software, or a Web-based security scan service.
10. Be informed! Know about problematic email attachments and high-risk files or links. Learn about common Internet hoaxes and real threats before forwarding security alerts.

SUMMARY

Computer and network security has many facets, and this overview touches on just a few of the security issues encountered by PCs and mobile devices connected to the Internet. Understanding security begins with a proactive stance and commitment to ongoing vigilance. Some of the best advice is just common sense, but as threats become more sophisticated and pervasive, a higher level of attention is warranted by all, and especially for those with broader organizational responsibilities for IT.

REFERENCES

- Albanesius, Chloe. (2012, April 24). 20 Percent of Macs infected with Windows malware. Retrieved September 28, 2013, from <http://www.pc当地.com/article2/0,2817,2403463,00.asp>.
- Anderson, Nate (2013, March 24). How I became a password cracker. Retrieved May 1, 2013, from <http://arstechnica.com/security/2013/03/how-i-became-a-password-cracker/2/>
- Bright, Peter. (2011a, March 22). How operation b107 decapitated the Rustock botnet. Retrieved June 11, 2013, from <http://arstechnica.com/information-technology/2011/03/how-operation-b107-decapitated-the-rustock-botnet/>
- Bright, Peter. (2011b, March 29). Rustock repercussions: Spam down by a third, at least for now. Retrieved June 11, 2013, from <http://arstechnica.com/security/2011/03/rustock-repercussions-spam-down-by-a-third-at-least-for-now/>
- Brodkin, Jon. (2013, June 3). The secret to online safety: Lies, random characters, and a password manager. Retrieved June 19, 2013, from <http://arstechnica.com/information-technology/2013/06/the-secret-to-online-safety-lies-random-characters-and-a-password-manager/>
- CalTech Information Technology Services. (2003, January 6). NetBIOS blocked at campus border. Retrieved August 1, 2003, from <http://www.its.caltech.edu/its/security/policies/netbios-block.shtml>.
- Canavan, J. (2007, February 13). W32.Swen.A@mm. Retrieved June 19, 2013, from http://www.symantec.com/security_response/writeup.jsp?docid=2003-091812-2709-99.
- Cheng, Jacqui. (2013, March 21). Apple follows Google, Facebook, and others with two-step authentication. Retrieved June 19, 2013, from <http://arstechnica.com/apple/2013/03/apple-follows-google-facebook-and-others-with-two-step-authentication/>
- Chien, E. (2007, February 13). Nimda mass mailing worm. Retrieved June 9, 2013, from http://www.symantec.com/security_response/writeup.jsp?docid=2001-091816-3508-99.
- Chien, E. (2014, April 14). Heartbleed poses risk to clients and the Internet of things. Retrieved May 10, 2014, from <http://www.symantec.com/connect/blogs/heartbleed-poses-risk-clients-and-internet-things>.
- Cyberthreat forecast for 2012. (2011). Retrieved June 13, 2013, from <http://www.kaspersky.com/images/Kaspersky%20report-10-134377.pdf>.
- Dilger, Daniel Eran. (2013, May 14). Mobile malware exploding, but only for Android. Retrieved June 17, 2013, from <http://appleinsider.com/articles/13/05/14/mobile-malware-exploding-but-only-for-android>.
- Dragani, R. (2013, December 6). Stolen password analysis exposes foolish choices. Retrieved December 12, 2013, from <http://www.technewsworld.com/story/79574.html>.
- Eddy, W. M. (2006, December). Defenses against TCP SYN flooding attacks. Retrieved September 18, 2013, from http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-4/syn_flooding_attacks.html.
- Even, Loras R. (2000, July 12). Honey pot systems explained. Retrieved June 17, 2013, from <http://www.sans.org/security-resources/idfaq/honeypot3.php>.
- Griffith, Eric. (2011, November 29). Password protection: How to create strong passwords. Retrieved June 17, 2013, from <http://www.pc当地.com/article2/0,2817,2368484,00.asp>.
- Grimes, R. A. (2001). *Malicious mobile code*. Cambridge, MA: O'Reilly and Associates.

- The Heartbleed bug. (2014, April). Retrieved May 10, 2014, from <http://heartbleed.com/>
- Heise Security. (2008, January 15). Quantity of malware booms. Retrieved May 1, 2008, from <http://www.heise-online.co.uk/security/Quantity-of-malware-booms-/news/101764>.
- Internet security report. (2013, April). Retrieved January 31, 2014, from http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v18_2012_21291018.en-us.pdf.
- Kaplenk, J. (1999). *UNIX system administrator's interactive workbook*. Upper Saddle River, NJ: Prentice-Hall.
- Kerner, Sean Michael. (2013, September 13). Apple updates Mac OS X 10.8.5 for security, stability. Retrieved September 28, 2013, from <http://www.eweek.com/security/apple-updates-mac-os-x-10.8.5-for-security-stability.html>.
- Klosowsk, Thorin. (2013, January 7). How Web sites vary prices based on your information (and what you can do about it). Retrieved September 26, 2013, from <http://lifehacker.com/5973689/how-web-sites-vary-prices-based-on-your-information-and-what-you-can-do-about-it>.
- Lynn, Samara. (2010, September 7). Ten tips for public Wi-Fi hotspot security. Retrieved June 17, 2013, from <http://www.pcmag.com/article2/0,2817,2368802,00.asp>.
- McCall, T. (2007, December 17). Gartner survey shows phishing attacks escalated in 2007; more than \$3 billion lost to these attacks. Retrieved June 18, 2013, from <http://www.gartner.com/it/page.jsp?id=565125>.
- MessageLabs March 2011 intelligence report. (2011, March). Retrieved June 11, 2013, from http://www.messagelabs.com/mlireport/MLI_2011_03_March_Final-EN.pdf.
- Mills, Chris. (2013, April 23). Five simple ways to keep your Android malware-free. Retrieved June 17, 2013, from <http://gizmodo.com/5995254/five-simple-ways-to-keep-your-android-malware-free>.
- Morris, Chris. (2012, October 24). The 25 most popular passwords of 2012. Retrieved June 18, 2013, from <http://games.yahoo.com/blogs/plugged-in/25-most-popular-passwords-2012-164015152.html>.
- Parsons, June, & Oja, Dan. (2012). *NP on Computer Concepts 2013 Comprehensive*. Boston, MA: Cengage Course Technology.
- Pogue, David. (2012, February 28). Use it better: 8 Alternatives to the hated Captcha. Retrieved June 19, 2013, from <http://www.scientificamerican.com/article.cfm?id=pogue-8-alternatives-to-hated-captcha>.
- Raiu, Costin, & Emm, David. (2012, December 5). Kaspersky Security Bulletin 2012. Malware evolution. Retrieved June 13, 2013, from <http://www.securelist.com/en/analysis/204792254/>
- Rubenking, Neil. J. (2010, April 7). Analyst's View: PDF—Pretty Dangerous Format? Retrieved May 1, 2014, from <http://www.pcmag.com/article2/0,2817,2362356,00.asp>.
- Rubenking, Neil J. (2013, March 20). 7 Signs you've got malware. Retrieved June 13, 2013, from <http://www.pcmag.com/article2/0%2C2817%2C2416788%2C00.asp>.
- Security TechCenter. (2014, May 1). Security update for Internet Explorer (2965111). Retrieved May 11, 2014, from <https://technet.microsoft.com/library/security/ms14-021>.
- Sophos, Inc. (2002, May 1). Melissa worm author sentenced to 20 months. Retrieved June 18, 2013, from http://www.sophos.com/pressoffice/news/articles/2002/05/pr_uk_20020501smith.html.

132 Internet Technologies and Information Services

- Sophos, Inc. (2006, June 8). JDBGMR hoax. Retrieved June 18, 2013, from <http://www.sophos.com/eln-us/threat-center/threat-analyses/hoaxes/virus-hoax/jdbgmgr.aspx>.
- Strange, A. (2014, April 8). Widespread encryption bug, Heartbleed, can capture your passwords. Retrieved May 10, 2014, from <http://mashable.com/2014/04/08/major-security-encryption-bug-heartbleed/>
- Symantec Internet security threat report. (2011). Retrieved January 31, 2014, from http://www.symantec.com/threatreport/topic.jsp?id=threatreport&aid=malicious_code_trends_report.
- Trend Micro. (2008, January). Malware today and mail server security. In *A Trend Micro White Paper*. Retrieved May 1, 2008, from <http://www.emediausa.com/FM/GetFile.aspx?id=8541>.
- Whitwam, Ryan. (2013, June 7). Kaspersky researchers discover most advanced Android malware yet. Retrieved June 17, 2013 from <http://www.androidpolice.com/2013/06/07/kaspersky-researchers-discover-most-advanced-android-malware-yet/>
- Windows rootkit overview. (2006, March). Retrieved January 31, 2014, from <http://www.symantec.com/avcenter/reference/windows.rootkit.overview.pdf>.
- Yaneza, J. L., Thiemann, T., Drake, C., Oliver, J., Sancho, D., Hacquebord, F., et al. (2008). 2007 threat report and 2008 threat and technology forecast. Retrieved June 18, 2013, from http://us.trendmicro.com/imperia/md/content/us/pdf/threats/securitylibrary/tre_threat_report.pdf.



Part

2

Building the Web

Part 2 focuses on the technology of Web content production and includes chapters on general principles of Web design and graphic formats for the Web, HTML, the use of CSS for presentation control, CMSs, and the applications of scripting to creating more dynamic and interactive sites. The section concludes with an introduction to XML and related technologies.

This page intentionally left blank

Web Design and Graphics

Web design is a broad topic with many facets. It encompasses technical and organizational skills, an understanding of graphic design, and an appreciation of the ways human factors engineering can be applied to enhancing the user experience.

WEB DESIGN OVERVIEW

Web publishers and site owners now have a number of options to deploy their site, from building it manually to employing any one of many CMSs or free hosting services. Regardless of the approach used for its implementation, those in charge of websites must still be cognizant of the principles of Web design to produce aesthetically pleasing and functional sites. Any discussion of Web design must acknowledge and appreciate that there are both aesthetic and technological considerations and that the skills and talents needed for each are quite different. Web designers need artistic sensibilities and a technical foundation to implement their designs. They must also be aware of how the shift to smartphones and tablets is changing both design and user behaviors. Designers need an understanding of the limitations of smaller, mobile device screens, as well as how HTML, CSS, and scripting expand or limit design possibilities. In addition, the results of Web design efforts are potentially affected by the server that hosts the site and by the viewing software on a particular device. The browser, the types of display device, the OS of the host, and the specific hardware configurations in use all potentially impact design choices and results.

On the surface, Web design can seem straightforward and somewhat formulaic. An overview of the design process could be summarized in the following “10 step plan”:

1. Define the mission of the site, identify the audience it is to serve, and assess their needs;
2. Examine other websites to develop an appreciation and understanding of the design approaches of similar organizations;
3. Develop an initial site plan and get the needed organizational “buy in” from those in a position to approve or veto the site;
4. Explore the site-hosting environment to determine technology options and preferences;
5. Develop and/or evaluate the content the site is to present;
6. Create various schematics such as wireframes and “story board” the site to model the structure of the site and the interconnections among the different main pages;
7. Develop a visual metaphor or concept for the branding and rapid identification of all pages as being part of an intellectual whole (an image of the site, called a “comp” is often created at this stage);
8. Create or acquire the needed graphics and a graphic layout version prototype;
9. Create a prototype, test for accessibility and usability on different devices, and then modify based on user feedback or focus groups;
10. Launch and monitor the site for scalability, usability, and actual use using tools such as Google Analytics.

Although each step might seem straightforward, the reality is that each presents a number of issues and details to address, and requires different kinds of expertise and input. For instance, designers interested in developing an in-house dynamic database-driven site must work closely with the technical experts to determine the server-side add-ons it would require as well as produce the scripts that draw content from the database. Those using an open-source CMS such as Drupal must have the expertise needed for its implementation. Even a site on a hosted Web publishing service that handles the technical requirements and provides templates will require attention to the principles of good design.

Successful designs require knowledge of not just the content presented and the technologies of the Web but also an understanding of visual arts. Because the intention of many websites is to inform an audience, knowledge of the pedagogy of online learning environments also contributes to successful design. Web design employs many different techniques and requires a vast array of knowledge, skills, and talents, which is why large-scale organizational Web development is not typically a single-person enterprise but instead often delegated to a design team.

Design Teams for Website Development

When the Web first burst on the Internet scene, many organizations responded by either outsourcing their design needs to professional designers or

by looking for in-house expertise to handle their Web development. Limited resources, coupled with the newness of the Web, often led organizations to try to manage development internally with existing personnel. Often a single person, usually someone in IT, was designated as the "Webmaster" (a term that has fallen out of favor today) with full responsibility for the launch of the website. Typically, these individuals were already involved with other Internet technologies such as running FTP or gopher services. Their expertise was often in UNIX server administration and programming, not necessarily graphic design. Consequently, much of the early Web tended to be graphically unsophisticated, utilizing textual designs on the browser's default gray background. Today, there might still be a "Webmaster" or a single individual who administers the server and manages the programming role, but that person is no longer usually the sole designer and proprietor of the website. The desirability of including graphic artists, designers, and content experts, coupled with the use of Web authoring and content management programs, has reduced the need for the technical expert to handle all aspects of Web design and implementation.

The design team has many advantages over a single person. Not only is it difficult to find a single individual with all the needed skills and talents, but a team approach offers differing perspectives on what makes an effective design. The design team typically still needs a technical expert to deal with issues related to the hosting environment or with the Web programming needed for dynamic Web designs or the CMS. The server OS, the directory structure employed, and the other server-side programs available are also the responsibility of the technical expert. For instance, a design plan based on Microsoft's Active Server Page (ASP) technology requires additional server-side programs to function, which may or may not be available. In addition, the IT administrator can arrange FTP access to the server or to the CMS for those who need it to upload content.

Content expertise is necessary to create and evaluate the material presented. This expertise could come from a single person or committee, but CMSs make it possible to distribute content control and facilitate collaboration across a larger group within the organization. The content experts develop new materials, evaluate them, and make selection decisions about digitization of existing materials and prepare them for Web delivery. Designers must also make content format decisions; for instance, the choice of HTML, PDF, or other proprietary formats such as Microsoft Word or Excel significantly affects subsequent access to Web content.

The graphic artist/designer is a key player in the team. Almost anyone can learn HTML, but the creative side of Web design requires the unique training and talents of design experts. A full exploration of the graphic arts is outside the scope of this text. Just as learning the rules of a game does not make everyone expert players, learning about design principles certainly does not make everyone an artist. However, a discussion of graphic design guiding principles may at least lead to a conceptual understanding of these issues and an appreciation of the critical role of text fonts, color, and visual layout in successful designs.

The Web team might also include someone with legal expertise to address questions about intellectual property rights, develop or vet consent forms for images, and comment on accessibility compliance. Engagement of higher

management is also important to ensure that the site is developing appropriately within the mission of the organization and is consistent with organizational policies. Management should designate who has the authority to make final decisions about the implementation of the design.

Site Mission and Audience

In many ways, a Web development project is similar to planning a database. Large database projects usually begin with systems analysis to address fundamental questions and create system models. This process includes the distinct steps of needs analysis, data dictionary development, and system modeling using dataflow diagrams and/or entity-relationship (ER) diagrams. As in database design, where the first step is to identify what kinds of questions the database is expected to answer, the Web designer begins by seeking answers to the questions of who is expected to use the site and why. Who is the intended primary audience? Is the site primarily for an internal audience or more for public relations (PR) and directed to potential external constituents? Is the site attempting to sell something, or is it mostly informational? How will the design accommodate both mobile and desktop users? How does it connect to other out-reach activities such as the use of social media?

These questions often do not have a single or simple answer and the various answers result in potentially conflicting design imperatives. For instance, the goals of a PR department may prioritize branding at the expense of usability. Consider the example of a large academic library website. It is part of a larger organization, such as a university or college, which may impose its own design mandates. The mission of the site is multifaceted and includes delivery of information and services as well as a possible information literacy/teaching role. The audience is diverse and includes undergraduates, graduate students, faculty, staff, and often the public. Each of these groups brings very distinct needs, expectations, and levels of experience to their use of a library website. The audience has expectations not only regarding what information is available but also regarding ease of use; Google has dramatically influenced user expectations in both areas. One “Google effect” is that many users accustomed to the simplicity of its interface become frustrated by the more complex information environment presented by library websites. However, complexity is not always just the result of the interface design but is a direct result of the many functions library websites must perform as well as the huge array of databases and services they present, some of which are not really controlled by the library, such as access to eBooks or magazines that require users to have OverDrive or Zinio accounts. Library websites are both a marketing and PR tool—a way to provide information about the system, its hours, and locations, services, and staff listings. It must also serve as a gateway to the myriad databases, indexes, and catalogs that are available. Library site designs are often complicated further by the fact that the databases they offer are often heterogeneous information “islands,” each with its own separate interface. There is interest in applying federated search or metasearch approaches to address this problem, but these single-search solutions are not yet perfected. Given the diversity

of both the audience and the various content sources libraries must manage, it is not surprising that a design for such a site involves many tradeoffs that rarely satisfy all users.

The ability to see a site from each of these different users' perspectives is imperative. In library sites, the content expert team members are generally the librarians themselves. Assuming that library users are the main audience of their efforts, designers do need to be careful about employing jargon. Many terms and acronyms are instantly recognizable to librarians as efficiently naming a service or product may not serve as meaningful labels to others; for instance, terms such as OPAC, WorldCat, JSTOR, or EBSCOhost all would require further explanatory text or alternate labels.

Library Web designers often find themselves engaged in the same debates about their roles that take place in their other activities, such as providing what the public *wants* vs. what *should be* in a collection, or the sometimes-conflicting roles of teacher vs. mediator/provider. The library website is not only a gateway to information, but also a teaching tool, providing opportunities to inform users and enhance their level of information literacy. With differing missions, another design debate has emerged between those who wish to emulate the simplicity of the Google search interface as worthy of emulation and those who are concerned about the "dumbing down" of search systems so they no longer serve the needs of all its users. As the website is an extension of the library itself, it is not surprising that these debates find their way into Web design, and they will remain issues once the site is complete.

In the initial design stages, it can be very informative to ask the various constituencies to evaluate the designs of similar organizations. This process can be made more rigorous with an evaluative matrix that allows numeric ratings of various design elements such as visual appeal, color schemes, layout, navigation, and content organization. Such a process will highlight the often-conflicting assessments of different user groups. Data collection can reveal trends regarding what elements most users find effective and attractive as well as which elements they do not, resulting in a design that incorporates the "best practices" of many similarly focused sites.

General Design Guidelines

Much of Web design literature is directed to a business audience, but the general principles described are applicable to all types of Web development, including that done by libraries. One cautionary note about general design guidelines is that they can be contradictory and are constantly evolving to accommodate new technology and new thinking about usability. For instance, two suggestions are "make links predictable" and "make screens as simple as possible." However, in order to make links predictable, you may need to add extra explanatory text, which violates the "make it simple" rule. Another guideline is to require a consistent look and feel to every page on a site; yet, there may be instances where more idiosyncratic designs for a particular page better serves a specific function or niche audience. Ultimately, final design choices reflect the tradeoffs inherent in resolving the difficulties of presenting complex information environments while attempting simultaneously to keep

things simple. Guidelines provide best practices, not fixed rules, and it is with this caveat these principles of Web design are considered.

Key Elements of Web Design

As you were always told, first impressions matter. The site home page is the entry point for most users, and it is the initial hook (McClelland, Eismann, & Stone, 2000). Regardless of the wealth of content within the site, the entry page must engage users and help them understand both where they are and why they should stay and look around; it shapes the user impression and experience with the site. If the main page is too busy, has poor navigation, or does not make clear the nature of the organization that it represents, it will not be successful. Frustrated users will abandon a poorly designed site and choose an alternate path to the information they are seeking. Krug (2000) emphasizes this with his first law of usability, which he states is “Don’t Make Me Think!” This does not mean a site cannot have multiple options presented or be somewhat complex in its offerings. It means that good designs use clear and simple jargon-free language, make navigation and links obvious, and provide search functions that are easy to identify and use. It also implies that cognitive overload can be an issue in designs that are overly textual or complex; it is generally better to break up long pages into more cognitively manageable “chunks.”

A successful entry page needs to accomplish multiple goals. It should (1) “brand” the site with a meaningful visual metaphor, (2) have a clear purpose beyond a simple “welcome” message; it should clearly present useful content and/or information about the organization it represents, (3) orient users to where they are on the site, (4) provide excellent navigation, and (5) be stylistically and holistically consistent. Further, all these must be accomplished with the understanding that most users glance at a page to quickly scan for the first link that seems vaguely related to what they are seeking (Krug, 2000). User studies of eye tracking reveal this scan often takes the form of an F-shaped pattern; users quickly scan down the left side and across the top and middle areas of the page, so those areas should be given special consideration when positioning essential text. Other specific features expected in all designs include a search-within-the-site option that is clearly identified, clickable elements that are visually obvious, the last modified date, and contact information for the owner of the site. This last item is unfortunately frequently missing; it is surprising that many websites do not include basic contact information or telephone numbers in a convenient, easy-to-find location for those who are simply using the site to find a contact person.

To grab the visitor’s attention, designers are often tempted to try to impress users with the latest “cool” technique. This can be striking, but it is important to consider what value is added by the special effect, especially if there is potential for some to find it annoying as opposed to cool, and they can quickly become dated. For instance, when Flash animations first became popular, many entry pages used them but they quickly lost their charm.

Another consideration is, as they say in real estate, “location, location, location,” which on the Web is the URL itself. The URL is not only used by the

browser and bookmarked by the user, but it also appears in many documents such as letterhead, business cards, and brochures. A very long URL can be problematic for these publications and for the user who must enter it from a printed source. URLs that result from database-driven websites are particularly challenging; they are potentially long and do not end with a meaningful file name. One simple way to shorten a URL is to use one of the default entry page names of **welcome** or **index** with the **htm** or **html** extension. When these names are used, they can be omitted from the URL. If the designer has control of the URL, keeping it short and meaningful is ideal. The URL assigned is usually predetermined by the organization, but there are strategies that allow it to be shortened or modified if needed. These usually involve some type of link resolver or the use of a name alias that performs a redirect to the site. One approach is to use a link resolver service such as Google's URL shortener (<http://goo.gl>) or TinyURL (<http://tinyurl.com/>). A long URL is submitted to the service that maps the true URL to a shortened one in the form <http://service.com/>. name. However, such an alias hides the true location of the site, which raises branding issues for organizations or can result in abuses such as creating innocuous-appearing links to shock sites (sites intended to shock or offend) or to mask sites with malicious code.

A related issue is that URLs that are very similar to yours can create confusion for the user or result in a potentially embarrassing situation. The URL “whitehouse.com,” formerly a pornography site and now a commercial ad site, was often confused with the official “whitehouse.gov” site. Purchasing and holding unused but similar URLs is a common strategy to prevent this situation; when initially registering their domain name, organizations can register all the variants of the URL that are still available.

A key design concept is site consistency; each interrelated page should have the same look and feel. Text styles should be consistent and navigation, search features, and logos consistently placed within the page. The look and feel of the site comes back to the idea of branding mentioned earlier. Branding is a key design element that helps users always know what organization the site represents, no matter what page they come to in a search. Developing this visual metaphor for the site allows users immediately to recognize your brand identity by the logos, color schemes, and textual cues. The use of templates and style sheets can greatly assist in creating and maintaining visual consistency throughout a site.

A common goal for many business sites, especially those using the site to sell a product or earn income through ads, is the creation of a “sticky” site, that is, one that retains a user for some time each session. This is not a universal goal; the click-through business model does not expect users to dwell on that site but instead use it as a means to find some other location. Search engines recognize that most users are destined for other sites and do not expect them to linger on their site for long, but as much of their income comes from advertising, they do have an interest in retaining users long enough to present advertising and sponsored links. However, for many business models “stickiness” is a goal: the longer a user is on the site, the better the chance that they will find something of interest that might lead to a sale. As search engines expand their services to become Web portals, they care more about extending the time a user stays on their site. Many search engines have become full-fledged portals, that

is, a site defined as your starting point for all subsequent Web activities. Such full service portals seek to keep users by offering search along with newsfeeds, email, Web hosting, shopping, messaging, or other services. The idea of stickiness has both positive and negative aspects. On the positive side, it can mean that a site is so visually interesting and useful that users tend to stay on the site for an extended time. However, sites can achieve stickiness through techniques that do not embody good design, such as not allowing back button navigation and continually reloading themselves into the same or a new browser window, or deploying multiple popup windows before allowing the user to move on. A well-designed, informative site retains users not because they cannot escape but because they find it a worthwhile and engaging experience.

A number of guidelines qualify as design myths, or at least overstatements. Heuristics such as “The Rule of Seven” or “The 3 Click Rule” may be well entrenched but are worth revisiting. The rule of seven suggested people can best handle at most seven categories of information; the three-click rule requires all points of content should be available within three mouse clicks. Wisniewski (2008) suggests that neither matters as long as categories are appropriate to the content and the clicks to the content have feedback to show users they are on the right path. This latter idea has been called the “information scent” that provides cues to the user that the trail they are on is leading to the desired location (Nielsen, 2003).

The notion that people always want more selection is often embedded within Web design thinking. A counter trend in design is to enforce the “KISS” rule—keep it simple. This view is predicated on the belief that too many choices can be confusing, so it is better to emphasize simplicity. This view appears in other contexts—for consumers in the marketplace, it is described as the “tyranny of choice” (Schwartz, 2005). Schwartz examines the paradox of a consumer-driven society where choice is highly valued juxtaposed with the evidence that too many choices can make even simple decisions overwhelmingly complex. Applied to Web design, the thinking is that pages with too many choices results in a design that does not present any of them very well and that needlessly confuses the user with endless options. Designs that try to do everything for everybody are “Swiss Army knife” designs. It may sound great to have one tool that is not just a knife but also a can opener, bottle opener, corkscrew, file, screwdriver, toothpick, and saw, until you actually try to use it for these diverse functions. In Web design, the implication is that a site that tries to perform many functions for a diverse audience is inherently problematic. However, this is exactly what libraries must often accomplish in their designs. In addition, there are exemplars of sites that handle site complexity well; Amazon is a multifocus site that presents many diverse products and services. Large-scale library websites must accomplish a similarly wide breadth of sources and services, offering many choices out of necessity. These sites benefit from strong organizational thinking to compartmentalize the diverse choices in way that effectively leads the user through the maze of possibilities. It seems that the KISS rule, like all broad generalizations, can be overstated; its application depends on the mission of the site. Simplicity is easier to prioritize in designs focused on a more specialized audience or functional niche. For libraries, such niche pages might take the form of a topical LibGuide (discussed further in the chapter on CMSs) that target a specific student assignment or library service. Mobile devices have introduced additional new challenges when designers must present

large sites with multiple choices; an alternative to a complex site with many options is to build specialized, dedicated apps for specific functions or activities.

User Experience Design

Studies of user preferences should inform design practice. User-centered design is the application of a deep understanding of user desires, needs, and preferences throughout the design process of any consumer product. In the context of Web design, this idea is embodied within the field of useromics, which is the application of human factors engineering to the user experience. It has given rise to what is known as *user experience design* (UXD, sometimes referred to as UED). It puts the user front and center in design process. This broad field takes in many related topics that are situated at the intersection of humans and technology, from studying the eye movement of users while they view a site to various forms of usability testing. A report of the User Interface Engineering Task Force offers some rather counterintuitive findings that highlight the difference between print and Web publishing. For instance, it reports that the use of white space, which is generally important in print designs, can hurt Web usability. However, design trends that encourage simplicity and use of white space would take exception with that finding. The task force also found that users are more successful with information seeking when following longer, more descriptive links compared to shorter, less informative ones. This is further evidence that explanatory text can be helpful. It also concluded that users do not always hate having to go below the “fold” at the bottom of the screen, so requiring some vertical scrolling to see additional content is not always a design flaw. However, it does seem that users consistently dislike horizontal scrolling or having to resize pages to be readable on mobile devices, and designers should avoid making that necessary.

The design field is evolving as the nature of the Web, user behaviors, and the devices they use continue to change. Armed with more data on the user experience and behaviors, some designers and information architects are challenging many accepted design assumptions. Peter Morville (2005) summarizes three key “mantras” that should guide Web design (p. 104):

1. “You are not the user.” He emphasizes the need to understand the user experience through using data-driven approaches such as field studies and search log analysis.
2. “The experience is the brand.” He suggests a shift from a preoccupation with site image to the site experience itself; this implies that attractive sites cannot compensate for a negative user experience.
3. “You can’t control the experience.” Simply stated, users do not always engage the site in the ways you have anticipated, and your expectations often are not aligned with the user experience.

Web 2.0 and Design

Web 2.0 thinking has definitely influenced Web design. There is even some speculation that Web 2.0 tools may eliminate Web design as an autonomous,

stand-alone process performed separately from simple content management. Even with user-friendly tools such as Adobe Dreamweaver or a CMS such as Drupal, creating and managing a website and keeping it current is a labor-intensive activity. Web 2.0 technologies provide an alternate avenue to traditional Web publishing for content delivery and promotion. Instead of a traditional website, blogs are available with free accounts on WordPress.com or Blogger.com. Images or video can be pulled into the blog content from Flickr or YouTube, and an RSS subscribe option can be added through a service such as Google's FeedBurner. These combined activities result in a Web-delivered, targeted content stream that is easy to update with little of the fuss of a full design implementation. This is clearly appealing to many who do not have the resources to manage a full Web presence or a local CMS. Although many larger organizations are utilizing similar strategies, some still manage a locally implemented Web design. One hybrid solution is to maintain a site by distributing the content update function throughout the organization with a content management program. Expertise with design, HTML, CSS, and scripting is still needed within the organization, but those without these skills can still participate in revising content.

CMSs are Web 2.0 technologies that provide access to a WYSIWYG (what you see is what you get) editor for content updates. The Web editor allows source code manipulation if needed; however, HTML knowledge is not required. The edited updates are actually posted to a backend database table; the Web page with the new content is delivered through styled templates and scripts that pull the new content from the database. One such system used by some libraries is the open-source CKEditor (<http://www.ckeditor.com>). Another popular open-source CMS is Drupal (<http://drupal.org/>), a feature-rich platform that utilizes PHP and MySQL technologies to build powerful and flexible websites. CMSs and other hosting options are discussed in Chapter 11.

Web 2.0 thinking has influenced design by reinforcing the mantra to "keep it as simple as possible." As discussed in the previous section, simplicity may result in fewer site features but with the benefit of ease of use and less user frustration. The Google search page is an exemplar of this "less is more" design paradigm. Norman (2007) argues that complexity in design is not inherently bad; people often accept the tradeoff of more complexity for the benefit of added useful features. It seems the bottom line is that while simplicity should be considered in design, it must be adapted to, and balanced with, the mission of the site.

In addition to visual simplicity and a centered design with a good deal of white space, Kroski (2007) highlights other important characteristics of Web 2.0 design, such as more interactivity; interface development utilizing XML and scripting technologies; the use of overlays with maps and other "mashup" approaches; and more collaborative, socialized designs. Socialization of the site includes opportunities for user commentary and reviews, RSS feeds and subscriptions, and "send-to-friend" options for site content. Another interactive feature is the use of links that result in pull-down text boxes or call-outs instead of loading a new page. While all these features extend the design possibilities, designer Jakob Nielsen adds a note of caution, warning that although Web 2.0 is the "latest fashion," designers should still heed established

principles of good design that emphasize usability more than trendiness (*BBC News*, 2007).

Mobile 2.0 and Design

Mobile began with simple cell phones, but the smartphone has led to the convergence of the Web with mobile, which some refer to as “mobile 2.0.” Just as Web 2.0 was the realization that the Web was the platform, mobile 2.0 anticipates that mobile is becoming the primary context of the Web, making the browser a key mobile app. However, traditional Web access gave users a relatively uniform experience regardless of platform—PC or Mac, IE or Firefox, the experience was quite similar. This is not the case with mobile where there are many platforms and carriers, each of which can change the user experience. Browsers need to support AJAX for rich interactions, but AJAX is JavaScript combined with XML, and JavaScript is problematic for many mobile browsers. As discussed in Chapter 10, JavaScript requires client-side processing, and running scripts adds processing demands that can affect device performance and reduce its battery life more quickly.

The shift from the desktop to smaller mobile devices changes the design parameters for developers as we move toward the post-PC world. Smartphone shipments have overtaken those of the desktop PC and more and more people access the Web with a mobile device exclusively. However, most websites were designed for the PC with its much larger screen, faster processor, and higher capacity memory. The task for many website developers is to “mobilize” the site, not only to ensure it looks good and functions as expected on the device but also to optimize the experience for the mobile user. As Layon (2012) notes, devices are not mobile—people are. These mobile users benefit from app and browser-accessible geolocation capabilities that allow them to access location-relevant content. Mobile users are busy, sometimes more distracted (or relaxed), and often have pressing information needs that can make them more task centered than the typical desktop user. Not only are the behaviors and tasks of mobile users often different, but the available bandwidth and screen real estate is limited and the finger activated touch screen interface can make navigation or following links problematic. Not all mobile browsers can handle scripts, CSS, or have needed plugins such as Flash or a PDF reader. These issues combine to create new design imperatives for the developer to consider.

Designers have a number of options to accommodate mobile users, ranging from doing nothing to building dedicated mobile sites or specialized apps (Layon, 2012). Doing nothing creates the risk of not serving a potentially huge audience. One option is to view a page using a transcoding site to convert the page by stripping out videos and multimedia and then shrinking and compressing elements into a series of smaller pages; an example is the Google transcoder at <http://www.google.com/gwt/n> (Kyrnin, 2006). However, these automated approaches do not necessarily optimize the mobile design or user experience. There are CSS options for mobilizing sites but this approach is not universally supported across platforms. Designers may opt to create a simple page for mobile and redirect users from the main site to this alternate page.

Creating multiple pages for various devices is also possible, but it can become an increasingly complicated solution that does not scale well for large sites. The other main strategy is to employ a responsive design that can accommodate different devices.

Responsive Design

Mobile developers are increasingly considering a design solution that allows a site to work well across many platforms; this design strategy has come to be called *responsive design*. As described by Ethan Marcotte (2011) who coined the term, responsive design employs three technical elements: fluid grids, flexible images, and CSS3 media queries. Fluid grids allow for the rearrangement of page elements to fit the screen, flexible images can dynamically resize to accommodate the display limitations, and media queries can determine the exact nature of the display environment. Examples of how these technologies can be implemented within HTML and CSS will be highlighted in Chapters 8 and 9. Two good resources on responsive designs and their implementation are from Kadlec (2013) and Frain (2012).

The proponents of responsive design refer to it as “one-Web” that utilizes a single set of coding for all devices, but it does have limitations. Responsive images can fit neatly into a small display, but they may not scale up well when viewed on a larger screen (Marquis, 2012). Reformatting content for different devices may be an excellent approach, but it is not the optimal solution for all mobile tasks. Those tasks provide the “mobile context” that should be considered by the designer. For example, a user of a banking site might have goals that are very task centered and a dedicated mobile site or app is needed to deliver the optimal individual experience (van Vuuren, 2012).

Tips for Mobile Designers

Some general suggestions emerge from the discussion earlier on mobile design. A few of these items are self-explanatory, and others will be examined with examples provided in the HTML or CSS chapters.

- Rethink the design; do not just recycle an existing site.
- Rethink site hierarchy—many levels are hard to navigate and are not desirable in a mobile design.
- Consider screen real estate and the appearance of the site in both landscape and portrait mode viewing as well as on different devices.
- No frames, ever!
- Tables: although they are not used for layout in traditional design, they have come up again in some mobile designs as a work around or pragmatic solution to a specific design issue.
- Take advantage of geolocation features and maps when they are potentially useful.
- Be aware of image size issues and file size download time.

- Improve readability with appropriate fonts and line spacing.
- Test your design. Options for testing can be a simple iframe window sized to mimic a specific screen size, an emulator, simulator, or the actual devices.

Apps vs. Website

An alternative to forcing the user to view your content or service within a browser is to build dedicated apps to perform a needed function. Apps have a number of advantages:

- They are optimized for the platform and device in use.
- They are designed for one primary function and thus better suited to the task-specific information behavior of the mobile user than a general website.
- They typically can utilize geolocation information to provide location-relevant information.

SITE PLANNING

After the basic questions about the nature of the site are addressed and the initial content decisions made, it is time to plan and model the site.

Schematics

To start, it is helpful to create a schematic of the anticipated site structure. There are many techniques employed for this design phase, ranging from a simple sketch on paper to more formal “storyboarding,” a technique used in the visual and dramatic arts to understand the story to be told visually with scene posters. Film storyboarding is a very involved process that visualizes the scenes and characters in the order they will appear; similarly, the “scenes” of a site are the main pages within it. Web schematics are functionally similar to database modeling techniques that utilize dataflow diagrams to identify processes and/or ER diagrams. Web schematics help the designer identify what the site is about and how the elements are interrelated. Initial views might emphasize just the main pages and their relationships, but other more detailed maps might show the full site hierarchy. This initial drawing or blueprint might begin with a rough sketch using the “butcher paper method” of design, such as that shown in Figure 7.1. A more formal schematic is a *wireframe*. Wireframes are a stripped down visual representation of the site showing the page architecture. They can show the overall layout, a content inventory, and various page elements such as the location of headers, footers, or images. In addition, element behaviors can be described (Definition & usage—Wireframe, 2009). Wireframes are created for the home page, each major subpage, or portal page, and for templates. These then lead to a more formal representation of any site hierarchies showing how individual pages connect. Figure 7.2 is an example of a more formal site map.

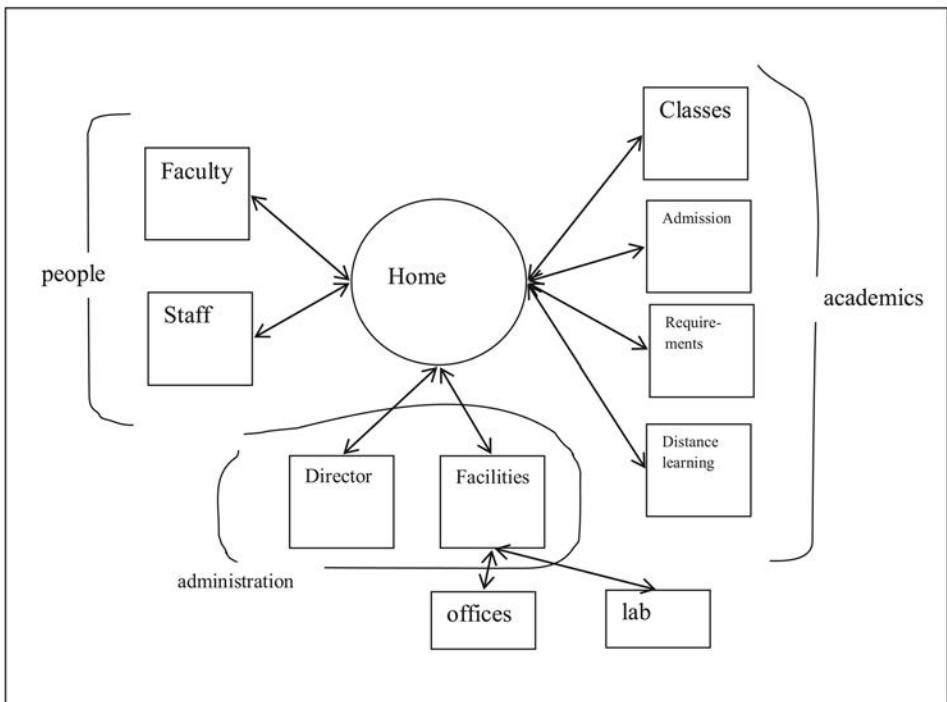


Figure 7.1 A butcher paper sketch of the main pages in a design.

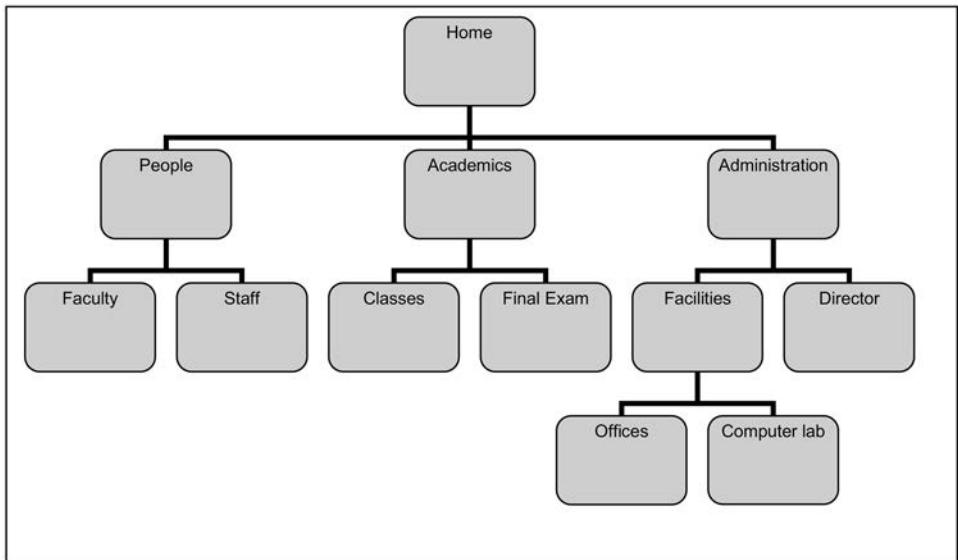


Figure 7.2 A formal site map showing the site hierarchy.

Layout

Cultural influences play a role in layout preferences. Westerners read from left to right and from top to bottom, so our eyes tend to gravitate initially to the upper left. This tendency is supported by eye tracking studies of people viewing Web pages. As mentioned earlier, a study by the Nielsen Norman Group found that users specifically view a page in an F-shaped pattern—a horizontal glance across the top, followed by a second horizontal glance at about the middle of the screen, and finally a vertical glance down the left side (Nielsen, 2006). These findings reveal that (1) users do not read a page thoroughly and (2) essential content should be placed in one of these specific optimal areas on the screen.

There are many tools for creating a layout mockup from a wireframe schematic; for example, both Adobe Fireworks and Adobe Photoshop are often used for these (Jones, 2010). Although Fireworks was specifically built for Web designers, many designers still use Photoshop to create a visual prototype, which can be a quick and easy way to visualize the design to be implemented (Yates, 2013). The use of a graphics program at this stage to create an image of the desired final look of the site, called a “comp,” has several advantages. Photoshop was often the program of choice for creating individual graphical elements such as buttons, and shadow effects using program tools; further, many designers and graphic artists already had substantial experience with that program from past print design work. A limitation of graphics programs is that they create bitmaps that result in pixel-based outputs, but these fixed size mockups are still useful for exploring color options, fonts, textures, and moods. Twitter designer Samantha Warren (2013) suggests Photoshop can be used effectively to create Style Tiles, described as “a design deliverable consisting of fonts, colors and interface elements that communicate the essence of a visual brand for the web” that can “help form a common visual language between the designers and the stakeholders” (<http://styletile.es/>). The fixed measures utilized in the comp image can be converted later into flexible equivalents, as discussed in the section on responsive design. Layout is influenced by the viewing medium, which for the Web are usually screens and not a printed page. Layout is complicated further by the variety and unpredictability of the various types of devices used to view the site as well as the various screen sizes and resolutions in use. The desktop computer browser has been the main way people view Web content, but mobile and handheld devices are increasingly being used to access the Web instead of the standard browser. Such devices often require alternate views of the content that do not utilize style sheets or scripts. Responsive design is one solution, but anticipating these parameters remains a design challenge. Although media queries can attempt to match a page version to the viewing device, some design decisions reflect a best guess about the “typical” viewing environment of most users, which is becoming harder to do because smartphone and tablet devices now come in many sizes and resolutions. Screen sizes are stated as a diagonal measure given in inches. There are “flip” style smartphones that have small 2.2 inch screens, and some traditional flat screen models are as small as 2.8 inches. The early versions of the game changing iPhone had 3.5 inch screens. The trend is toward larger screens, and smartphone sizes of 4 to 6 inches are common. Tablet screen sizes range from 7 to 12 inches. Resolutions, measured as the number of pixels per inch,

have improved dramatically in recent years. The earlier iPhone 3GS resolution of 480 x 320 was relatively low compared to the newer iPhone 5 resolution of 640 x 1146 (css3html5, 2012). Resolutions are trending ever higher and full HD resolutions of 1920 x 1080 or higher are common for mobile devices, and resolutions as high as 4096 x 2160 are being developed. However, these very high resolutions will require more processing power than many current devices have now (Boxall, 2013). Many designs use a multifaceted “container” view of the document. Such a design might begin with a layout template that specifies four distinct areas within the page as shown in Figure 7.3. For a mobile device, these four areas would be better viewed in a layout where these areas appear linearly, as shown in Figure 7.4. Both these general layouts are being modified by designers to adapt to anticipated user needs and the newer “looks” available in CMS templates. For example, many large-scale sites have replaced the large content area shown in Figure 7.3 with slideshows or other media elements along with multiple boxes for added content such as news and social media streams. Complex mobile sites may elect to not present an initial content area in the linear layout shown in Figure 7.4 and replace it with a set of links to the major subpages needed for most mobile users. Figure 7.5 shows an example of an academic library mobile site that uses this approach.

There are many possible templates, and each has potential issues to consider. Yates (2013) referencing Ethan Marcotte, points out that:

Thanks to Ethan’s concepts, web designers have come to realize that they should be thinking from the content outwards, not the page boundaries inwards (though it should be noted that this approach isn’t compulsory). We don’t know how big a web page is, so we need to design our content to fit into whatever boundaries it’s faced with. Think of web content as being like a liquid; capable of being poured into all manner of vessels.

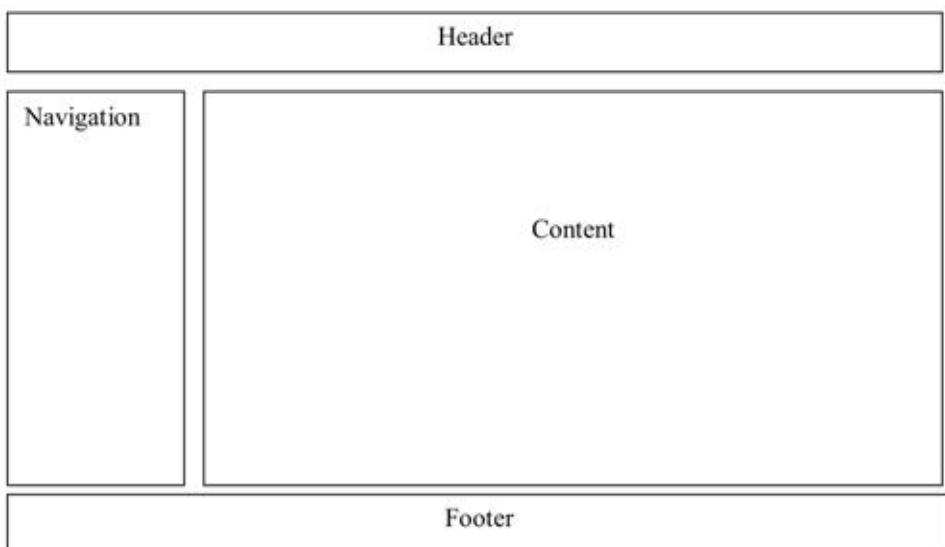


Figure 7.3 A traditional design template for desktop viewing.



Figure 7.4 A layout template for mobile devices.

The style-coding details used to achieve these effects and some of the different “boxes” shown in the figures mentioned earlier are discussed in the chapter on CSS. As an aside, note that the CSS float declaration, which determines the box location, is not handled in the same fashion by all browsers. In addition, having a footer appear at the bottom of the *page* is not the same as making it appear at the bottom of the *screen*. If the content box above is full, the footer appears at the bottom of the page, but controlling the location of the footer is problematic if the content area is less than a full screen. In many earlier Web designs, templates often used HTML tables or framesets to accomplish this layout. Framesets present problems that make them inappropriate for modern designs. Tables have resurfaced as a pragmatic occasional layout solution in some mobile designs or for possible CSS problems in some browsers (Marquette, 2010). However, using tables for layout creates accessibility problems, and generally there are better layout options available in CSS.

Site Organization

Site navigation and site organization go hand in hand. Both of these interrelated topics are reflected in the site model documented in the wireframes and overall site schematic. The overarching goal of site organization and navigation is to maximize information access and minimize the number of clicks needed to get to the desired information. Design heuristics refer to a “three-click” rule, which suggests that everything someone needs should be within three clicks. This is not always realistic for large-scale sites, which would require too much navigational overhead in each page. Nonetheless, it is worthwhile to consider how much effort users have to expend to get to the content they need. Although these topics are interrelated, site organization is considered first, as these organizational decisions largely determine the internal navigational strategies for the site.

In the world of the printed book, content is thematically related and the author imposes the narrative order. Navigation is then straightforward because the expectation is that topics are encountered sequentially; chapter numbers



Figure 7.5 A screenshot of an academic library mobile site that provides a link list to major features needed by a mobile audience (used with permission).

and page numbers along with an index provide nonlinear access to content. However, websites represent many “stories” that are not presented in a linear fashion. Website designers often confront the same information organization problems they would when building subject directories. Making links to *everything* from the home page is rarely possible. Designers therefore consider creating navigational aids such as branching structures that fan out from the home page, menu-based hierarchies to organize the site, a site index available from each page, a visual site map, multiple pages layered within a single screen with “tabs” to access each, a “site search” appliance, or various combinations thereof. As discussed earlier, mobile versions of complex sites often present little or no narrative content beyond a linear list of links that allow users to jump directly to that section of the site, making decisions of major topic areas and their description critical to the mobile presentation. Each solution entails tradeoffs, and no solution is perfect.

Site hierarchies and drop-down or pull-out menus are two popular strategies but present design issues to consider. For example, menu systems that

depend on JavaScript for submenus cause problems for ADA accessibility and on mobile devices. Hierarchies require the formation of categories and have the advantage of organizing knowledge according to some predetermined scheme. However, this approach assumes a high degree of match between the categories and terms used by the creator and user. In addition, hierarchy navigation can be inefficient; if users are uncertain about the correct entry point, they must drill down in a branch to discover if they are on the right track. The addition of a search function can circumvent this problem.

Hypertext

Content has driven the success of the Web, but it is the power of hypertext, the ability to create links within a narrative to other parts of the document or to other related information sources, that uniquely defines the Web and differentiates it from other content delivery technologies. All Web designs are (or should be) created to take advantage of the power of hypertext. The Web as a whole is nonhierarchical and nonlinear in use. It empowers users to create their own paths through a knowledgebase that is largely unconstrained by a narrative order imposed by the content creator. Hypertext, coupled with the fact that the viewing medium is typically a screen and not paper, significantly influences the publishing process for the Web author. A large textual document can be presented in a continuous flow that the viewer can scroll through, but large blocks of text are usually broken up into discrete but interlinking pages that are graphically enhanced. Because the content may be divided among many separate files or Web locations, each of which can be indexed by search engines, the Web author must assume that users do not always enter the narrative from a logical starting point. This means that good navigation is paramount to enable users to see how the pages they have found fit into a broader intellectual framework. The selection and creation of meaningful hyperlinks within a work, as well as related external content, add value to the Web publication. In addition, the link structure and its associated anchor text is used by some search engines to assess page relevance, which makes the creation of meaningful links that are associated with appropriate text an important activity.

Links to other places in the same document are *local* and are references to HTML named anchors within that same document. Links to outside sources (i.e., to different files or sites) are *external*. The default browser rendering of a link is to show it in blue text with underlining; this default can be changed, or styles can be applied to links that alter the appearance of a link when hovered over or followed. One consequence of the common use of the default rendering for links is that other text styles are not always recognized as hyperlinks, and conversely, other uses of underlining or blue text for nonlink text can be confusing because most users assume the text represents a link.

Navigation

The hypertext, nonlinear nature of Web content makes good, consistent navigational tools essential, and hypertext is both part of the problem and the solution. Hyperlinks within the page empower users to create their own narrative

and path through the content, but they can also be a detriment as users can lose focus and lose their way. Navigational aids allow users to return to their initial path after exploring new information avenues. Such navigational aids can be textual, image based, or both. The key is that the navigational links should be consistent in location and function on every page. Search engine results can land a user in a page that is quite distant from the initial home page making good navigational aids imperative so users can quickly see how to navigate from that entry point to the main page of the site.

Navigation links with simple words such as “top” are informative enough, but words such as “back” are not very meaningful. It is more helpful to have less ambiguous and more descriptive navigation links. As an aside, a page script may force a reload of the same page, effectively disabling the use of the browser “back” button; although site navigation should not depend on browser functionalities alone, it is not good practice to force a page reload unless there is a compelling need to prevent back navigation. The application of styles or scripts can make textual navigation links more visually animated and noticeable. Navigation can be associated with images by using either the whole image or selected image areas to provide multiple links to create an image map. Roll-over scripts that change in appearance when hovered over or clicked can also activate image links, but CSS approaches that make use of styled lists of links are displacing these image-based navigation techniques.

TYPOGRAPHY AND FONTS

The varied Internet protocols accommodate the transmission of textual information. Text formats are those where every byte value is associated with a character in the ASCII text-encoding scheme. HTML files are a special form of simple text file where certain characters are reserved for markup; a huge amount of the content available on the Web is in this form. Text files are just containers of character data, but how these characters appear when printed on a page or screen is their *typography*. Typography is expressed as different fonts and sizes. Although related, typography and font are not synonymous; typography refers to a typeface and may include a whole family of fonts, whereas a font defines a specific style. In addition, the text appearance conveys or adds meaning to the message. Fonts have “personality” and can be formal or informal, serious or frivolous. Fonts can use bold or all caps for emphasis, italics for highlighting, and light or heavy weights. Just as text decoration reveals something about the nature of the text to the reader, search engines also make use of text decoration in determining term weight in an index.

There is some jargon related to typography that identifies important attributes essential to the control of font properties in style sheets. Fonts have a style, identified by the name of the typeface. Font size is measured in points or pixels. A point is a printer’s measure that is about 1/72nd of an inch. A larger unit, the pica, measures 12 points. Pixel designation can also be used to specify size for screen display but will produce variable sizes depending on the resolution setting of the display. Given the common resolutions in use on standard desktop screens, 10-, 12-, or 14-point fonts are often best, and anything smaller than 10 points is not recommended for Web pages on those displays. Fonts rest on a baseline and the amount of space between two adjacent

Serif font (Times New Roman)	Sans serif font (Verdana)
T H Y I P i l p	T H Y I P i l p
H T P I Y i l p	H T P I Y i l p

Figure 7.6 Serif font Times Roman on left is compared to sans serif font Verdana on the right.

baselines is the *leading*; increasing the leading gives more white space between lines and can improve screen readability.

Font styles can be serif or sans serif. Serif font verticals end with a right-angled stroke that gives a “flourish” to the stroke. Sans serif fonts lack these right angle finishes to the vertical. Examples of each type of font are compared in Figure 7.6. Serif fonts have definite appeal in printed materials, but sans serif are generally preferred for display on a monitor. Web readability studies at the Software Usability Research Laboratory (SUR) find that while there are not significant readability differences in serif compared to sans serif fonts at the same resolution, there are differences among font types (Bernard, Mills, Peterson, & Storrer, 2001). They found that Arial is much preferred for screen reading when compared to Times New Roman; they also found that as a group, older adults prefer sans serif fonts.

Font alignment on a page requires controlling margins (top, bottom, left, and right) as well as text justification, which can be left, right, or full. These distances can be specified in styles as absolutes or in a relative fashion. Font alignment is particularly important in navigation areas, and many sites use styled lists as navigation menus. Designer Jakob Nielsen reports that right-aligned menu lists impede readability; a ragged left edge to the text interferes with a quick scan of the menu (Nielsen, 2008). Font positioning and other font properties are discussed as part of typographical control in CSS style sheets.

GRAPHICS AND COLOR IN DESIGN

The capability of the Web to deliver and display images within a page greatly enhances the richness of Internet content. Successful Web designers must master enough of this complex topic to understand and use the various graphic formats and color schemes, to know what formats are best for what application, and to understand the trade-offs between resolution choices and reasonable file size for optimal display. This section focuses on the technical side of graphics as opposed to the more artistic and creative aspects of image creation and use; these topics are best left to those with the requisite training and artistic talent and are not explored here. McClelland, Eismann, and Stone (2000) provide a number of practical tips for working with graphics. Understanding graphics is also needed in part to ensure that the graphic format chosen is one that a browser can display; the common formats that most browsers handle are GIF, JPG, and PNG. The Scalable Vector Graphic (SVG) format is not supported in browsers such as IE 8 and some Android versions,

but it is gaining acceptance as these older programs are being replaced. However, these Web displayable formats are not good formats for retaining all the advanced features possible within sophisticated graphics programs. For all features to be available for future image edits, the image must be saved in the proprietary format first; a second “save as” in the appropriate format creates a version for the Web.

As part of this discussion, it should be noted that traditional use of images are not always the best design solution. The mobile Web and responsive designs have led many designers to focus on *performance-based design*, which requires rethinking both how graphics are created and how they are used. As discussed in the chapter on the stateless HTTP protocol, multiple images can mean multiple http requests for large files, which can be a huge performance bottleneck, especially for the mobile Web. Performance-based solutions include using CSS for graphic effects such as *sprite files*, where different parts of a single image can appear with shifting background positions, using fonts for logos, and SVG (Coyer, 2009). Thus, a single image file is downloaded but can give the appearance of multiple images, depending on the CSS coding.

A few of these techniques are explored in Chapter 9.

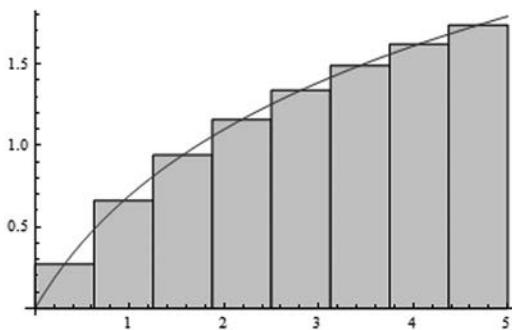


Figure 7.7 The area under a curve can be approximated by summing the areas of rectangles that can be drawn under it. Here eight rectangles are used.

of images. Sampling techniques are analogous to approximating the area under a curve by summing the area of rectangles drawn within it. The larger the number of rectangles used, the better the approximation of the area and the smoother the curve they appear to form becomes. For instance, a better approximation of the area under a curve results if 24 rectangles are used compared to when eight rectangles are used. Figure 7.7 shows the curve with just eight rectangles used under the curve;

Analog to Digital

Converting analog data to digital requires the smooth curves of the analog data be approximated in digital form through data sampling. This idea was introduced in Chapter 1 in the context of sampling rates used to convert analog music to digital form based on Nyquist–Shannon Sampling Theorem (Shannon, 1949). Sampling issues also apply to the digitization

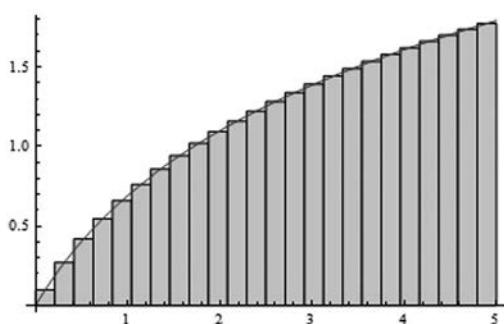


Figure 7.8 The area under the curve is more accurately estimated with 24 rectangles compared to the previous example.

Figure 7.8 shows the area estimated with 24 rectangles under the curve. For both examples, a consequence of this approach is that the edges of the curves appear “jagged” when the shape is approximated in this fashion. This relates to the aliasing effect, introduced in Chapter 1 and discussed again below.

Color Use

The Web has become much more visually attractive and graphically animated since the early days of text-intensive pages that used black text on default gray backgrounds. Certainly, aspects of color selection are personal, and preferences are quite subjective. However, color choice is more than a personal choice; the emotional and/or cultural influence of color should be considered as well. HTML allows for the use of hexadecimal codes to specify colors based on varying amounts of red, green, and blue. The amount of each is represented with an 8-bit number and can be expressed either as a 3-digit base 10 number (0–255) or a 2-digit hexadecimal one (0–ff). A number that is all zeros indicates the absence of color (black) while a maximum level of each is white (the full spectrum of white light). Color codes appear as an attribute in many HTML tags and style rules.

A few general guidelines about color are useful in this discussion of design principles. Borrowing from the Hippocratic Oath, the designer’s first priority should be to do no harm. Dramatic and busy background colors can be visually stimulating but also reduce contrast with the text, resulting in difficult-to-read pages; this also applies to background images that appear as a watermark. The priority should always be readability. Color choices should also be tempered with an understanding of how long visitors are expected to stay. A splash welcome page with minimal text may utilize bold colors that might be less appropriate on a page requiring more reading of text where bold color choices become visually fatiguing. One technique to troubleshoot color issues is to change the site pages to a grayscale background. By removing the color and replacing it with varying degrees of gray, you can test for contrast and identify colors that may interfere with readability.

Psychologists know that color can evoke an emotional response, and its use in Web design establishes a “mood.” Some color combinations are dissonant, and too many colors can create visual chaos. Graphic artists recommend choosing a dominant color first, and secondary colors later, where the role of the secondary color is to lessen or accentuate the dominant color choice. This is where the aesthetic sense and expertise of the graphic designer is important to the design team.

There is a cultural component to color responses; for instance, in the United States, different red and blue have a patriotic connotation. There are many such examples from different cultures; for instance, the color yellow is associated with cowardice in the United States but happiness in Egypt, success in India, grace in Japan, and wealth or power in China (Russo & Boor, 1993). Awareness of the symbolic power of color is needed when designing for a culturally diverse audience.

Color choice also becomes an accessibility issue as it relates to readability, especially because it is estimated that about 1 in 20 people have color perception disabilities. Various forms and degrees of color blindness can make it

difficult to distinguish between, for example, red and green. Color contrast issues make some text almost invisible against certain background colors and could make sections of an image map difficult to use. Graphics and color choices should therefore be tested on one of the many sites designed to reveal potential color use problems.

Image Symbolism

Images can represent ideas and content in many ways and, whether used for navigation or to supplement and enhance the content, are part of the appeal of Web publishing compared to earlier text-only Internet services. Some image uses are examples of iconography, which is an attempt to represent the shape or features of an object; a stick figure drawing of a person may not be very sophisticated but is nonetheless instantly recognizable. Ideographs are images that are not as literal but represent an abstract concept, such as the use of the skull and crossbones to indicate danger or death. Mnemonic symbols are ones that have a learned meaning; for instance, a hypertext link is a learned symbol. Much of the success of computer graphical user interfaces (GUIs) is due to the preference of most people for icons to represent commands pictorially. Many images used for navigation are simple buttons that contain descriptive text within them. Hover over and on-click changes further animate image links and indicate their function. Navigation images without explanatory text are potentially problematic because iconic representations may not be recognizable by all; in addition, recognition of ideograph or mnemonic symbols is culturally dependent and may not be universally understood. Alternate text provides additional description for image-based links and is necessary for ADA accessibility compliance.

Image Maps

An image map allows for different parts of an image to be associated with different URLs. Image maps can serve as a site map or as a navigation bar that appears on every page. There are two classes of image maps: server-side and client-side. Both treat a two-dimensional image as an x - y coordinate plane; different shapes are defined in the image by identifying their geometric coordinates. Each shape can then in turn be associated with a unique URL, complete with alternate text. These coordinates and URLs are stored in the map tag instructions located in a map file or the HTML files itself. As the names imply, server-side maps require a program on the server that can read the map instructions; client-side maps are handled exclusively by the browser. The HTML code associated with a server-side and client-side map is discussed in Chapter 8.

Hardware and Graphics

The acquisition of graphic images has become much easier with the development of the vast array of relatively inexpensive digital devices for capturing images. Early in the development of PCs, scanners were expensive, slow, and

cumbersome to configure and use but now there are many types of scanners available with fast USB 3 or FireWire connections. Digital cameras are built-in to most mobile devices and enable the easy capture of images and video for instant Web distribution via social media. Large high-resolution flat screen desktop displays and high-resolution color printers allow high-quality output for online viewing or hard copy production. Mobile screens are small but have increasingly high resolution. Because the hardware determines the quality of the input and output of graphics, a review of some basic hardware concepts and terminology precedes the discussion of graphics.

Most Web images are viewed on a screen, which is comprised of *pixels*, a contraction of the terms *picture* and *element* and abbreviated as “px.” A pixel is a single point of color on the screen; the resolution setting and the dot pitch of the monitor determine pixel size. The resolution setting of a display refers to two different measures: the vertical resolution is the number of rows of pixels on the screen, and the horizontal resolution is the number of intensity changes permitted across each row of pixels. The dot pitch refers to the distance between pixels measured in millimeters. The higher the resolution, the more pixels there are per square inch. For instance, a 15" monitor set to VGA (640 x 480) has about 53 pixels per linear inch. Even as resolution has improved desktop screen sizes have increased, and 53–100 pixels per inch is common on many displays. The smaller screens of most mobile devices can have higher pixel densities than many desktops; for example, Apple’s iPad Retina displays range from 220 to 326 pixels per inch, depending on the size of the screen.

For graphics intended to be viewed exclusively online, using very high resolutions is often a waste of effort because they are not rendered that sharply on the screen. Manipulating and displaying graphics also requires significant computer memory resources. High-end computers for graphics may have stand-alone graphics processors such as those from Nvidia. Even at the very lowest resolution setting of 640 x 480, there are 307,200 pixels displayed on the screen. If the color depth setting requires 3 bytes per pixel, about 1 MB of data must be delivered to refresh the screen. This was a challenge for the limited memory of the early PC, which was one of the reasons additional memory was added to video cards. Screen output resolution is controlled by the video controller settings in the OS, and many resolution and color depth settings are possible. When different resolution settings are selected, the correlation between image resolution and image size becomes apparent; as resolution increases, image size always decreases, and conversely, as resolution decreases, image size increases.

Graphic Types

The two main types of images created by graphics programs are *vector graphics* and *raster graphics*. In a raster graphic (or a *bitmap*), the shapes are approximated by filling in squares within a grid much like the example of using rectangles to determine an area under a curve. The grid is an *x-y* coordinate plane, and the more boxes in the grid, the better the appearance of the desired shape. However, whenever this type of image is magnified, the grid itself is also

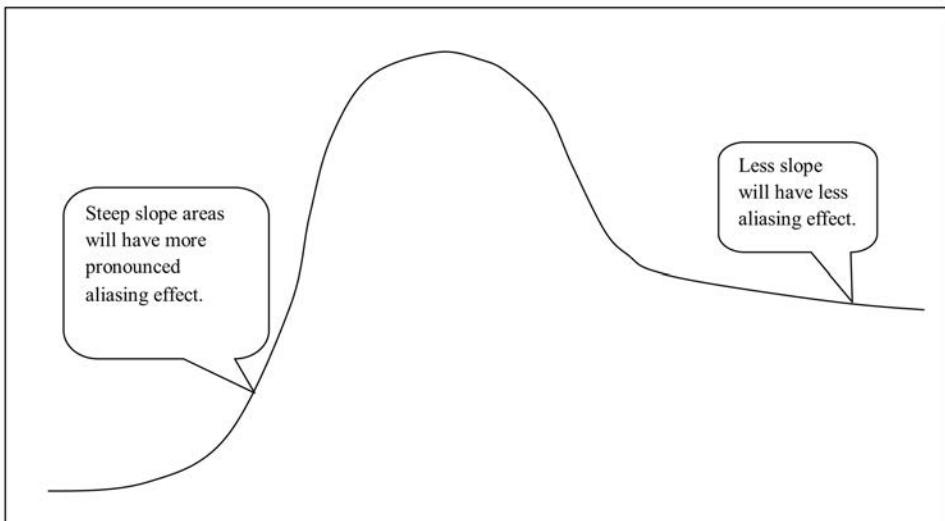


Figure 7.9 Aliasing will be more visually pronounced in the parts of this line with steeper slope compared to the flatter parts of the line.

expanded, and the jagged edges are revealed; it does not take much zoom magnification for an image to appear “pixilated.” The curved line shown in Figure 7.8 could be represented by superimposing it on a fine grid and filling in squares to approximate this curve. The creation of raster graphics always results in the edges of shapes having these slightly jagged edges. This effect is known as *aliasing*, which is a general phenomenon associated with the conversion of analog data to a digital representation. In Figure 7.9, this aliasing effect will be more visually pronounced in the parts of the curve with steeper slope than in the flatter parts of the line.

People are accustomed to seeing textual fonts rendered quite smoothly in most printed materials, so this phenomenon is especially noticeable when text is placed in raster images for use on the Web. Graphics programs compensate for this effect with an *anti-aliasing* feature. When anti-aliasing is enabled the program attempts to “smooth out” the jagged edges. The result is a slight fuzziness to

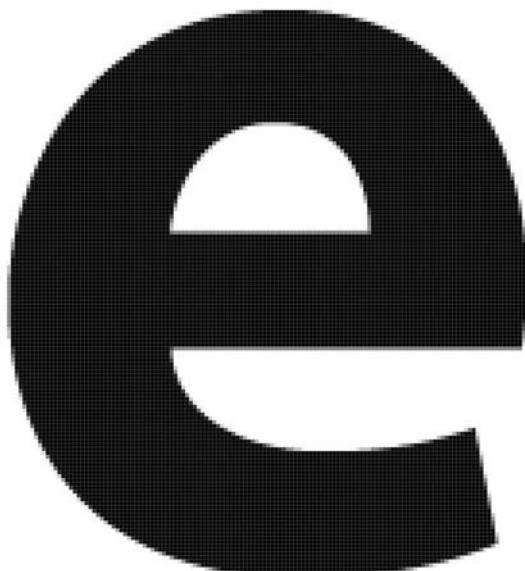


Figure 7.10 The magnified letter “e” created with antialiasing on.

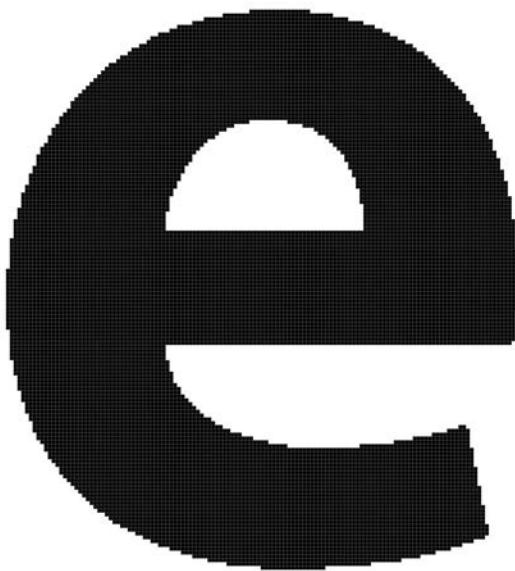


Figure 7.11 The magnified letter “e” created with no antialiasing.

shape is created from the new calculation. Both types of images have two essential properties: color depth and resolution. Resolution refers to the number of pixels used to create the image; color depth is the amount of digital data associated with each pixel to represent color.

the edges, making them appear smoother but also not quite as sharp. Figures 7.10 and 7.11 show this effect by magnifying the letter “e” created in a graphics program with and without anti-aliasing enabled.

The other main image type is the *vector image*. In a vector image, the shapes are mathematically defined by formulas to create shapes. For instance, a circle is defined in the formula (r^2) ; to make a circle shape larger or smaller, the program simply adjusts the value for the radius (r). This action therefore does not alter the quality of the final image as the new

Color Schemes

Graphics may be line drawings, bi-level images, or continuous tone images such as color photographs. Computer programs handle color images in different ways for different types of output; the three important color models available in most graphics programs are the RGB, HSL, and CMYK models.

All the colors of the spectrum can be represented by specifying the amounts of red, green, and blue mixed. This is the RGB model, and it is based on the way the human eye works to see a spectrum of color. White light is a mix of all these primary colors; this spectrum appears when light passes through a prism. In this model, the color black is represented as the absence of color and the color white with the maximum amounts of all three components. Computers use binary numbers for data representation, and the size of the number determines the number of unique “things” that can be represented; color depth relates to the amount of data associated with color representation. If 8 bits (8-bit color) are used to represent the array of possible colors, there are consequently only 256 colors in the palette ($2^8 = 256$). This was the color standard for VGA displays common with the early PC. If the system uses 16-bit color, 65,536 colors are possible; 24-bit color makes more than 16 million colors available (2^{24} or $256 \times 256 \times 256$). The RGB model is a 24-bit color model that uses 8 bits for the level of red, 8 bits for blue, and 8 bits for green, giving 24 bits total. This model is used in HTML and CSS to express color values for backgrounds, fonts,

and other elements. The computer sees these as 24-bit binary numbers, but the numbers are represented in decimal or hexadecimal notation by graphics programs and HTML-authoring programs. In decimal, these are expressed as three numbers that range from 0 to 255; in hex, they range from 0 to ff (ff in hex is 255 in decimal because $15 \times 16 + 15 \times 1 = 255$). Note that whenever all three color values are equal, a shade of gray results. When all three values are 0 the result is black (the absence of color); when they are all 255 (or ff) the result is white (the full value of each component of white light). By varying the amounts of each of these primary additive colors, any color in the spectrum can be represented.

There are constraints on how colors are rendered other than the number of colors theoretically possible in an image. For instance, graphics programs can utilize millions of colors, but if the computer display supports only 256 colors, that is what the user sees. Some graphic formats for the Web, such as **GIF**, use 8-bit color, so the file format selection determines the final color depth. A topic related to color depth issues is the “Web safe” palette, which goes back to the display standards when the Web first emerged. Because most PC displays at that time used the VGA standard with 256 colors, and the Apple computer of that era had color restrictions, a palette of 216 “safe” colors that were common to both the Mac and Windows platforms was adopted. The first browsers and Web graphics programs were designed to work with this smaller standard set of colors. When developers varied from this standard palette, a problem known as *dithering* could result, this happens when a nonsupported color is specified in an image but is unavailable to a browser. The browser would substitute a similar available color, often with poor results. Dithering compensated for the limited Web palette, but it could result in graphics with a blotchy appearance, especially in images with large color blocks of substituted color. Graphics programs usually have optimization features to control the effects of dithering, and programs often still include the smaller Web-safe palette option for simple Web graphics.

Another color model important to graphic artists is the HSL model, which stands for Hue, Saturation, and Lightness. People see hues of color, not mixes of the three additive primary colors described in the RGB model. Hue refers to the color’s place on the spectrum, represented on a scale of 0–255. Saturation values refer to the clarity of degrees of hue; lightness refers to how light or dark the color appears. Graphics programs can provide representations for any color using either RGB or HSL, as shown with the screen shot of the color selector Adobe Photoshop CS5 Extended version 12.1 in Figure 7.12 (note the check box that restricts the palette to the Web-safe colors only).

The final color model is the CMYK model, which creates colors by mixing various amounts of cyan, magenta, yellow, and black inks. This is the foundation for inkjet printing of color images; often the printer has separate ink tanks for these colors.

Resolution Issues

Generally, references to image resolution refer to the number of pixels per inch it contains. As noted previously, for images displayed on a computer

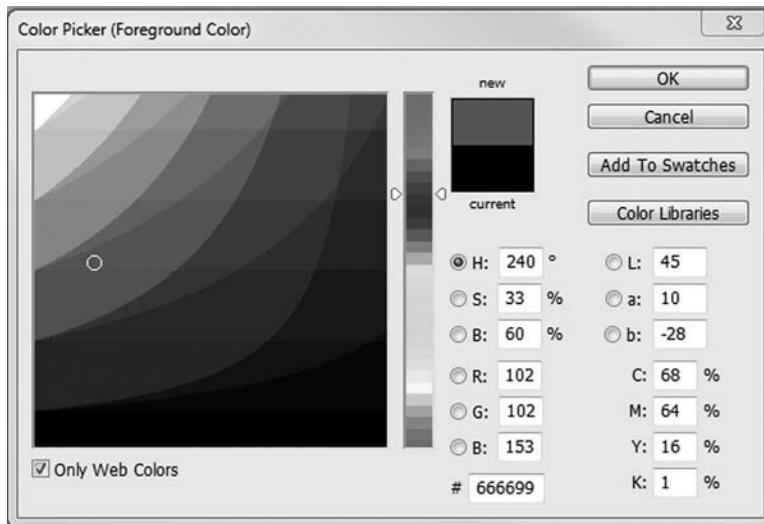


Figure 7.12 The color sector tool in Adobe Photoshop showing various color model representations (note select box option for “only Web colors”).

screen, the size of each pixel depends on the dot pitch of the monitor itself. However, the term *resolution* comes up in multiple, differing contexts. *Bit resolution* refers to the number of bits per pixel; *monitor resolution* is the number of image pixels on a screen, and *output resolution* refers to the DPI (dots per inch) of printed output. There are three key relationships to keep in mind regarding image resolution: (1) changing the image size always changes the resolution; (2) conversely, changing the image resolution always changes its size; and (3) increasing resolution can dramatically increase final file size. For Web presentation, it is best to keep file sizes as small as possible for the intended use of the graphic—this is especially important in mobile designs. Considering that the resolution of most computer displays is in the 70–120 DPI range, many images, especially block images such as banners or navigation bars, rarely need higher resolutions than that.

Graphic File Formats

Graphic artists can choose from a wide selection of graphics programs and formats. Some programs are quite basic and are available as free downloads, while other programs designed for the true graphic artist are more expensive and feature rich. Often graphics programs come bundled with a scanner or a digital camera and some cloud photo sites offer editing software along with data storage. These programs are generally adequate for basic image manipulation such as cropping or resizing images, adding text to an image, and selecting areas for new colors, but they may not have advanced features

such as magic wand or mask tools, filters, layers, and various special effects. Graphics programs have many output formats and a large number of “save as” options. Many of these formats use highly efficient compression algorithms to minimize file size. Some formats result in “lossless” compression, which means compression takes place without data loss that could affect the final image quality. Other formats result in “lossy” compression where there is some data loss and possible image degradation, especially with repeated saves.

One consideration when choosing a format is whether the image is to be shared among users who have access to the same graphics program or if some common exchange format is needed. If the image is destined to be part of a Web publication, it must be a format supported by Web browsers. Each graphics program has a default proprietary format that retains all the special features the program allows. It is always useful to keep a copy of any image work in that proprietary format for future editing. If the image is to be shared among users of different graphics programs, the TIFF (Aldus Tagged Image) format is considered a reasonable exchange format.

For Web publication, the most common formats are GIF, PNG, and JPG. The XML-based SVG format is receiving attention, but it is not yet widely supported by all browsers. The SWF Flash file format is another option for animated graphic content. These standard formats are described here.

GIF (pronounced “jiff”) is the Graphics Interchange Format and was originally developed by CompuServ for banner and other block color images. Although it was a proprietary format, it was so commonly appropriated by Web developers it became a de facto standard early in the history of the Web. The format uses LZW compression, which stands for Lempel-Ziv-Welch, the names of its developers. However, the patent for this compression technology was held by Unisys, so GIF use required licensing agreements to be arranged (Fahey & Brown, 1995). The Unisys patent for LZW then expired, so the GIF format was no longer restricted by it (Unisys, 2008). LZW is a lossless compression technology of the same form as zip-file compression. This algorithm associates longer bit strings with shorter codes stored in a “code book,” and this code can then substitute for other occurrences of the longer string in the file. This file format is available to Web publishers in all graphics programs. There are several versions of GIF, including 87a and 89a; the later allows for animated GIFs, where multiple still frames are saved in a single file container that can play back as a simple animation. GIF files are very compact and use only 8 bits per pixel. This keeps the file size small, but 8-bit color provides only a 256-color palette. GIF images display in interlaced data order, so images appear in full form on the screen and get gradually sharper. Transparent GIFS are also possible, which makes the background color of the page the background color of the image placed there.

PNG (pronounced “ping”) is the Portable Network Graphic, a newer standard specifically designed to be better adapted to the needs of Web publishing. It was intended to be a patent-free replacement for the GIF that would have no ownership or licensing issues. As with GIF, it uses lossless compression and supports interlacing while supporting better transparency capabilities and even smaller file sizes. Files are often 10–30 percent smaller than a similar GIF.

JPG or JPEG (pronounced “jay-peg”) is the Joint Photographic Experts Group format, a standard for continuous tone images such as photographs. The actual file container is the **JFIF** format (JPG File Interchange Format). JPG uses a lossy compression algorithm and is not suitable for many types of graphics such as line drawings. The lossy nature of this format means successive saves of the same image to JPG can result in degradation of image quality, so it is desirable to keep a copy of a photographic image in a proprietary format or an alternate lossless format such as TIFF, for future edits.

SVG, is a W3C recommended XML format used to define vector graphics (SVG Working group, 2003). It has a number of advantages; images do not lose quality when resized, can be printed at any resolution, can be indexed by search engines, can be animated, and they work well with JavaScript and CSS (Lumsden, 2012). Because of these properties, they are good choices for the Web, especially for high-definition displays. Development of this format began in 1999, but deployment has been slow due to past browser compatibility issues.

In addition to color and images, the ability to add multimedia provides another content option to designers, but just as with images, there are format issues. Creating and sharing video content has become incredibly easy compared to just a few years ago. The VHS video camera of that earlier time seems quite primitive compared to the smartphones or webcams of today. The Web designer seeking to deliver video content must be aware of the various formats and players needed on various devices and browsers. Often the best design option is to place the video content on a dedicated hosting service such as YouTube and embed a link to it. Specific sound and video content formats are explored in Chapter 13.

Using Images and Multimedia in Web Pages

Images and multimedia are incorporated into Web pages using HTML image or embed tags, and HTML5 offers the additional option of a video tag. As explored in Chapter 8, the image tag is an example of an empty element in HTML used to designate the place where an image is to appear in the page. Image tags should have alternate text, which is needed for accessibility because screen-reading programs depend on it to provide a description of the image. The alternate text is displayed when the cursor hovers over it, providing useful descriptive information to the user. The height and width attributes inform the browser how much screen area is needed for the incoming image so text can flow around it even as the image is downloading. More detail on other HTML image attributes are discussed in the next chapter.

From a design point of view, there are questions that should be asked about each image or multimedia element used:

- Does the image add value to the site?
- Is the image aesthetically pleasing?
- Given the limitations of the viewing medium, is the image file resolution appropriate?

- Is the image appropriately sized and proportional to the rest of the visual space?
- Is the image well balanced with the adjacent text?
- Is the image file size appropriate for an acceptable download time?
- How will it display in various viewing environments?

The answers to the first two questions are subjective, but the others are less so and best practices guidelines are available. Most graphics programs create new graphics at default resolutions settings higher than required for Web-based monitor display, especially for images with large blocks of uniform color. Scanned photographs may need to be resized or cropped to achieve images that are appropriately sized for the Web page. Finally, file size should be taken into account. If high-resolution photographs or images need to be made available, they should not be displayed in high resolution on the initial page but instead in a low-resolution version or as a thumbnail that is linked to the larger file for those who need the higher resolution version.

ACCESSIBILITY

Web designs are expected to meet accessibility standards, so an awareness of accessibility issues and their resolution should be part of all design plans. There is a huge amount of content being published on the Web without considering accessibility. The mandate for accessibility is not just a good design and ethical imperative—it is a legal issue as well. The Americans with Disabilities Act (ADA), passed in 1990, protected the access rights of the disabled and raised the level of general awareness of accessibility issues. The Rehabilitation Act Amendments of 1998, particularly Section 508, applied accessibility requirements to federal government electronic information and IT. Although this legislation was concerned primarily with access to federal information systems, recipients of funds under the Assistive Technology Act (AT Act) are expected to comply with section 508. The W3C Web Accessibility Initiative (WAI) (<http://www.w3.org/WAI/>) documents accessibility issues and best practices for Web design.

The numbers of people affected by disabilities is significant, and Web designs should accommodate their needs. The U.S. Census summary data for disabilities reported that as of 2010, 56.7 million people (18.7 percent of the population) had some level of disability and 38.3 million (12.6 percent of the population) had a severe disability (Brault, 2012). Specifically:

- About 12.3 million people aged six and over needed personal assistance with one or more activities of daily living (ADL) or instrumental activities of daily living (IADL).
- Approximately 8.1 million people 15 and older had difficulty seeing words and letters in ordinary newspaper print, including 2.0 million people who reported being unable to see.
- Approximately 6.7 million people reported difficulty grasping objects like a glass or pencil.

- An estimated 7.6 million people 15 and older had difficulty hearing a normal conversation, including approximately 1.1 million who reported severe difficulty in hearing. (Brault, 2012)

In addition to physical disabilities, census data also reveal that 15.2 million people (about 6.3 percent of the population) have some cognitive, mental, or emotional condition that interferes with their daily activities, representing everything from severe mental illness to various levels of learning disabilities.

Web designers must consider both what can be done as well as what should be done to accommodate the disabled in the design. These concerns are especially relevant in mobile designs where sight and hand mobility disabilities can be particularly problematic. First, accessibility awareness should be integrated into all stages of Web design and site implementation. One way to enhance accessibility is to utilize designs that incorporate multiple formats for content delivery. Some solutions are purely technical and involve some level of tradeoff between the goal of compliance and the desired functionalities of the site. Easy technical solutions include avoiding elements such as unlabeled graphics, inappropriate use of tables (including tables without summary information), having image maps as the only means of navigation, and menu options that are exclusively dependent on JavaScript. Screen reading programs, necessary for those with visual or mobility disabilities, have difficulty processing pages using noncompliant designs. These programs read across the page and cannot interpret tabular data or columns, and their use can result in a nonsensical reading of the page. In addition, they cannot provide meaningful information about graphics or image maps without the availability of descriptive alternate text.

Another accessibility issue occurs with dynamic Web content that is dependent on DHTML and various types of scripting. The Accessible Rich Internet Application is part of the WAI (WAI-ARIA) and it suggests making advanced site features more accessible by “providing information about user interface controls—such as expanding navigation bars—to assistive technology” (W3C, slide 24, 2007).

The use of a screen-reading program or a website that simulates one is an informative test for all Web designs. A related topic is the use of audio content without captioning or an alternate textual view of the content for those with hearing disabilities. The inclusion of navigation that is not exclusively dependent on the ability to use a mouse and an awareness of various assistive technologies can further assist those with mobility disabilities. Cognitive disabilities are exacerbated by the lack of clear navigation, overly complex presentations, the lack of illustrative nontextual materials, and the absence of descriptive links. Jargon that is not well described is even more of a potential issue for those with cognitive disabilities. Flickering or blinking text should be avoided as not just annoying but potentially dangerous to those susceptible to seizures.

All design prototypes should be tested for accessibility using one of the available Web-based services that examine the HTML of a submitted URL and generate a report identifying potential problems. The W3C WAI lists many of these tools at <http://www.w3.org/WAI/RC/tools/complete>. There are times when the value of a desired design feature might be balanced against the ideal of complete accessibility and result in its inclusion, but all such judgments should

at least be an informed and conscious decision based on tradeoffs among site needs, best practices, and available technologies.

USABILITY AND FINDABILITY

The discussion of design throughout this chapter emphasizes the creation of highly usable websites. Much of the design literature focuses on the issues surrounding how to make a site highly functional and usable to visitors who start engaging the site with the home page. However, as noted in other parts of this book, searching is one of the main activities of those using the Web. Users typically find content outside of the exploration of a single site through a search that leads directly to a particular page. This adds another dimension to the design process; not only do designers care about how well the site performs when it is used, but they must also consider the *findability* of the site. This does not mean the various principles of good design covered here are irrelevant; usability is still a priority, and the attention it has received has improved the quality of most websites over time. However, ensuring a site will appear within a set of pages retrieved by a search creates a new additional design imperative.

There are companies that offer search engine optimization (SEO) services, but Peter Morville (2005) describes a number of “findability hacks” the designer could consider. A primary strategy is determining the most common keywords your target audience use when searching for sites like yours and including them in your content as well as in metadata tags and in image alternate text. HTML framesets, JavaScript for menus, and dynamic HTML, however, all reduce the visibility of a site to search engines. Appropriate use of hypertext links also contributes to page placement within a ranked retrieval set by search engines that utilize link analysis.

Making a site not just findable, but highly visible within a search engines results page (SERP) has given rise to businesses specializing in SEO. Many of the techniques they employ to attempt to manipulate ranking are legitimate, but some are now viewed as “black hat” techniques that can result in a site being blacklisted by a search engine (William, 2013). Google offers advice to Webmasters on how to best optimize the findability of your site (Webmaster guidelines, 2013). Many of these suggestions are consistent with principles of good design highlighted in this chapter such as having useful content, clear navigation, a clear site hierarchy, and appropriate links. They also identify practices to avoid such as participating in link schemes, term stuffing, or recycling content from other sites. There are many good sources on the do’s and don’ts of SEO (Ward & French, 2013; Williams, 2103). This topic is also examined further in Chapter 15.

DESIGN AND THE LOWEST COMMON DENOMINATOR

A concluding topic is to reconsider the question of *who* the design is for. This discussion of design began by emphasizing the need to identify the audience, focusing primarily on site content. However, this question should also

be considered from the perspective of the intended groups' expected hardware and software preferences. Consideration of the likely viewing technologies used influences whether the design should be aimed at the "lowest common denominator" of the various technologies. Obviously, the designer would not expect any users are still using Netscape 1.0, but that said there are many different browsers and browser versions in use. Differences in display hardware can be especially problematic; although 15" screens set to 256 colors with 640 x 480 resolutions is no longer the norm, that combination is nonetheless a possibility. There are also differences between the Apple and Windows computers and between desktops and mobile platforms. Web browsers allow for a high level of user control of presentation and security settings, resulting in a viewing environment that is neither homogenous nor easy to anticipate. This means that it is incumbent on the designer to (1) try to accommodate these potential device differences in the design and (2) to test the results with multiple platforms, devices, browsers, and display settings. Tools such as Google Analytics can reveal much about the hardware in use by visitors to a site, providing data on the most common devices and their resolutions. The need to support all users highlights the importance of responsive design and the CSS techniques that enable different versions of the site to be presented to various display environments, but in the end, there are always design tradeoffs that require implementations not ideal for all viewing hardware or user agents.

SUMMARY

Websites were initially designed by IT personnel, but most designers now are graphic artists, whose talents and visual sensibilities are critical to developing an attractive, functional site. However, content development and technical functionality require a number of other expert skills. Because of the diverse knowledge and skills needed, a team approach with content experts working closely with graphic artists and technical experts is desirable. Even though not everyone may have a great deal of artistic talent, an understanding of general principles and design guidelines is attainable by all Web publishers. Similarly, although many artists may not be technical experts, an appreciation of the limitations and enhancements possible with HTML and Web programming can inform their designs. The initial phases of a typical design process include both content development and the creation of site schematics to envision site layout, organization, and navigation. Site appearance and readability issues require consideration of appropriate fonts, colors, and supporting graphics.

A basic understanding of graphics and the appropriate formats for them is needed for successful Web design and publishing. Graphic creation requires both technical and artistic skills, and even those with artistic training find graphics programs have a substantial learning curve. Most large-scale Web projects engage one or more graphic artists to ensure the graphics used enhance the site and create an appropriate balance of graphical and textual elements. Even if you are not the primary graphic designer, an understanding of

the format options available as well as the tradeoffs inherent with decisions about resolution, color depth, file formats, and resulting file size is important to all members of the Web project team.

Accessibility considerations and testing on multiple platforms with different display settings and browsers will reveal possible problems with the design. Designers must become familiar with accessibility issues and solutions, and test their work using one of the many accessibility evaluation tools. Finally, usability, and findability were considered in the context of design. Usability requires many strategies, from working with focus groups to reviewing data collected by Google Analytics. Findability uses various SEO techniques, such as the use of certain keywords, metadata, and managing links that can improve the chances a site will be found by users when searching.

REFERENCES

- BBC News. (2007, May 14). Web 2.0 'neglecting good design.' Retrieved July 1, 2013, from <http://news.bbc.co.uk/2/hi/technology/6653119.stm>.
- Bernard, M., Mills, M., Peterson, M., & Storrer, K. (2001). A comparison of popular online fonts: Which is best and when? *Usability News*, 3(2).
- Brault, Matthew W. (2012, July). Americans with disabilities: 2010: Household economic studies. Retrieved July 1, 2013, from <http://www.census.gov/prod/2012pubs/p70-131.pdf>.
- Coyier, Chris. (2009, October 24). CSS sprites: What they are, why they're cool, and how to use them. Retrieved July 1, 2013, from <http://css-tricks.com/css-sprites/>
- css3html5. (2012, November 24). List of tablet and smartphone resolutions and screen sizes. Retrieved May 12, 2014, from <http://css3html5help.com/list-of-tablet-and-smartphone-resolutions-and-screen-sizes/>
- Definition & usage—Wireframe. (2009, January 7). Retrieved June 21, 2013, from <http://web2usability.wordpress.com/2009/01/07/definition-usage-wireframe/>
- Fahey, M.J., & Brown, J.W. (1995). *Web publisher's design guide for Windows*. Scottsdale, AZ: Coriolis Group.
- Fling, Brian. (2009). *Mobile design and development* (1st ed.). Sebastopol, CA: O'Reilly.
- Frain, Ben. (2012). *Responsive Web design with HTML5 and CSS*. Birmingham, UK: Packt Publishing.
- Jones, Brandon. (2010, December 8). The Web designer's guide to comparing photoshop and fireworks. Retrieved July 1, 2013, from <http://webdesign.tutsplus.com/articles/general/the-web-designers-guide-to-comparingphotoshop-and-fireworks/>
- Kadlec, Tim. (2013). *Implementing responsive design: Building sites for an anywhere, everywhere web*. Berkeley, CA: New Riders.
- Kroski, E. (2007, April 2). Information design for the new Web. Retrieved October 1, 2007, from <http://infotangle.blogspot.com/2007/04/02/information-design-for-the-new-web>.
- Krug, Steve. (2006). *Don't make me think!: A common sense approach to Web usability* (2nd ed.). Berkeley, CA: New Riders Publishing.
- Kyrnin, Jennifer. (2006, November 22). Use the Google Transcoder to simplify your pages. Retrieved June 19, 2013, from <http://webdesign.about.com/b/2006/11/22/use-the-google-transcoder-to-simplify-your-pages.htm>.

- Layon, Kristofer. (2012). *Mobilizing web sites: Develop and design*. Berkeley, CA: Peachpit.
- Lumsden, Aaron. (2012, June 27). Getting started with scalable vector graphics (SVG). Retrieved June 21, 2013, from <http://webdesign.tutsplus.com/tutorials/htmlcss-tutorials/getting-started-with-scalable-vector-graphics-svg/>
- Marcotte, Ethan. (2010). Responsive Web design. Retrieved May 4, 2012, from <http://alistapart.com/article/responsive-web-design>.
- Marcotte, Ethan. (2011). *Responsive Web design*. New York: A Book Apart.
- Marquis, Mat. (2012, January 31). Responsive images: How they almost worked and what we need. Retrieved June 21, 2013, from <http://alistapart.com/article/responsive-images-how-they-almost-worked-and-what-we-need>.
- McClelland, D., Eismann, K., & Stone, T. (2000). *Web design studio secrets* (2nd ed.). Foster City, CA: IDG Books.
- Morville, P. (2005). *Ambient findability*. Sepastopol, CA: O'Reilly Media, Inc.
- Nielsen, J. (2003, June 30). Information foraging: Why Google makes people leave your site faster. Retrieved October 1, 2007, from <http://www.useit.com/alertbox/20030630.html>.
- Nielsen, J. (2008, April 28). Right-justified navigation menus impede scannability. Retrieved July 10, 2013, from <http://www.useit.com/alertbox/navigation-menu-alignment.html>.
- Nielsen, Jakob. (2006, April 17). F-shaped pattern for reading Web content. Retrieved July 9, 2013, from <http://www.nngroup.com/articles/f-shaped-pattern-reading-web-content/>
- Norman, D. (2007). Simplicity is highly overrated. Retrieved October 1, 2007, from http://www.jnd.org/dn.mss/simplicity_is_highly.html.
- Russo, P., & Boor, S. (1993, April 24–29). *How fluent is your interface? Designing for international users*. Paper presented at the INTERCHI '93, Amsterdam, The Netherlands.
- Schwartz, B. (2005). *The paradox of choice: Why more is less*. New York: Harper Perennial.
- Shannon, C. (1949). Communication in the presence of noise. *Proceedings of the Institute of Radio Engineers*, 37(1), 10–21.
- SVG Working Group. (2003, January 14). Scalable vector graphics (SVG) 1.1 specification: W3C Recommendation. Retrieved May 1, 2008, from <http://www.w3.org/TR/SVG>.
- Unisys. (2008). LZW patent information. Retrieved September 19, 2008, from http://www.unisys.com/about_unisys/lzw.
- van Vuuren, Cairn. (2012, November 18). Mobile apps: The trouble with using 'responsive design'. Retrieved July 1, 2013, from <http://www.forbes.com/sites/ciocentral/2012/11/18/mobile-apps-the-trouble-with-using-responsive-design/>
- W3C. (2007, July). WCAG 2.0 Web content accessibility guidelines update. Retrieved May 1, 2008, from <http://www.w3.org/WAI/EO/Drafts/wcag20pres/wcag2intro20070725.doc>.
- Ward, E., & French, G. (2013). Ultimate guide to link building: build backlinks, earn a higher search engine rank, increase the authority and popularity of your site. Irvine, CA: Entrepreneur Media.
- Warren, Samantha. (n.d.). Style tiles: A visual Web design process for clients & the responsive Web. Retrieved July 1, 2013, from <http://styletil.es/>
- Webmaster guidelines. (2013). Retrieved December 30, 2013, from <https://support.google.com/webmasters/answer/35769?hl=en>.
- Williams, A. (2013). *SEO 2013 and beyond: Search engine optimization will never be the same again!* Create Space Independent Publishing Platform. Self-published.

172 Internet Technologies and Information Services

- Wisniewski, J. (2008). The new rules of Web design. *Online*, 32(2), 55–57.
- Yates, Ian. (2013, May 13). Photoshop's role in a Web design workflow. Retrieved July 1, 2013, from <http://psd.tutsplus.com/articles/tools/photoshop-role-in-web-design/>

WEBSITES OF INTEREST

Annoying design at http://www.netmechanic.com/news/vol3/design_no14.htm.

Color blindness check at <http://www.vischeck.com/vischeck>.

Screen reader simulation at <http://www.Webaim.org/simulations/screenreader>.

Web Publishing with the Hypertext Markup Language

The rapid acceptance and overall success of the Web has been driven by the vast amount of ever-increasing useful content found there. The proliferation of Web pages was initially due to the minimal software and technical requirements of working with the HTML. Although Internet content exists in many formats, many of the resources accessed through browsing and Internet searching are still static, autonomous HTML documents. There are many options for content creation that do not require HTML skills, but in the end, the pages they produce are still based in HTML and CSS. The Web and HTML are closely intertwined technologies with a codependent relationship. The goal of this chapter is to develop an understanding of this underlying markup language. Chapter 11 explores options for developing such content without working directly in HTML.

An understanding of HTML is still important to most Web publishing ventures; even when content is dynamically generated from a database or CMS it is typically presented in the form of an HTML document created “on the fly.” Early Web pages were written with nothing more than a simple text editor and knowledge of the HTML, but now many sophisticated authoring programs provide a WYSIWYG view of the page as it is created. However, knowledge of HTML is still often required for various forms of problem solving and other direct code interventions. Even the users of CMSs can benefit from some familiarity with HTML coding. In learning HTML, the operative word is *language*, and as with any language, learning both the vocabulary and syntax is necessary to understand its structure, uses, and limitations.

This overview of HTML includes many of the commonly used markup tags used to structure and present documents, but it is not intended to be a comprehensive treatment of this extensive standard. There are many such comprehensive sources: Graham (1996) and Lemay (1997) both cover HTML, and

Powell (2003) includes coverage of XHTML as well as CSS and scripting. Frain (2012) provides an overview of how HTML5 and CSS3 can be used in responsive Web design. The goal of this chapter is to provide a general understanding of the structure of HTML along with specific uses of many common tags and to introduce useful new tags available in HTML5. HTML5 is gaining acceptance, but HTML 4.01 is still in common use, so much of this discussion and many of the examples provided are based on HTML 4.01. Differences between these two standards are highlighted both in the section on HTML5 and within the discussion of HTML 4.01 specifics. For more information on HTML, you should consult one of the sources mentioned earlier or any of the Web-based reference sources such as the W3Schools site (<http://www.w3schools.com/>). Also included is a brief discussion of the evolution of the HTML standard to an XML compliant form known as XHTML. Appendix 2 provides a brief discussion of a few of the issues that can arise for those who need to manage a Web presence on UNIX servers.

MARKUP LANGUAGES

The idea of “mark up” has been around as long as publishing itself. In the earlier, pre-Web context, markup referred to using margin notes to instruct printers and typesetters regarding fonts, point sizes, and styles associated with various structural parts of the document such as chapter headings. In a broad sense, markup is everything in a document that is not content. Markup standards for digital documents were developed early on in the history of computing and predate the Web. Documents have structural and semantic elements, and these can be described independently of the way elements should be displayed. For example, a header is a structural element that can be displayed in a variety of styles. A markup language is a way of embedding instructions to the displaying program that describe what the content means or how it should appear. Digital documents with markup applied are thus portable and adaptable to other media because programs can be created to interpret the markup instructions according to the lexicon and nesting rules.

There are two general approaches to markup for digital content. *Procedural* markup refers to the unique codes that various programs use to control a single way of presenting a document; for instance, the selection of 12-point, Times Roman font for a block of text. *Descriptive* (or generic markup) is a way to describe the purpose of the text rather than just the appearance. In this approach, content is separated from a specific style; for instance, certain text might be designated as a section heading, but the exact formatting of that type of text may vary in different renderings of the document. Procedural approaches were common but have several disadvantages: The author must invest significant effort and time on simply controlling the appearance of the content, global changes are harder to manage, software changes may mandate document style translation, and document interchange and consistency is limited.

The first standard for digital markup, developed by IBM in 1969, was General Markup Language (GML). Standard Generalized Markup Language (SGML) was developed as an ANSI standard in 1983 and it became an ISO standard in

1986. As the foundation of both a specific markup language of interest (HTML) and an important meta-language (XML), SGML is relevant to our introduction to markup.

SGML is a standard for creating and describing a markup language; as such, it is important to recognize that it is not a specific document language itself but is instead a meta-language. A DTD (Document Type Definition) is a specific markup language definition developed according to the rules of SGML. The DTD defines what the markup codes are and how they are to be processed. A DTD begins with a document type declaration and then defines the vocabulary and grammar of the new language by defining the permitted elements, their attributes, and the nesting rules for them. Specifically, the DTD describes the structure of the document in much the same way a database schema describes the types of information and the relationships among database fields; it provides a framework for document elements and serves as a document model. A document created according to the rules of some DTD is called an *instance document*. Documents are processed by a DTD reader. The DTD can be a separate file from the document or be integrated into the document file itself. By including the DTD with the document, any device with the appropriate reader can reference the DTD to print or display the document. A more detailed examination of the topic of DTDs is in Chapter 12, where it is examined in some detail and contrasted to the XML Schema Description Language (XSDL) as an alternate way to define a markup language. The key points about the relationship between SGML and HTML are:

- HTML 4.01 and earlier was derived from SGML.
- HTML 4.01 and earlier uses DTDs to define the elements and their uses.
- There are three variations of the HTML 4.01 DTD: strict, which does not include deprecated tags (tags that have been deemed unacceptable); transitional, which includes deprecated tags; and frameset, which includes everything from the transitional DTD plus support for framesets.
- The browser is the DTD parser; the DTD is “built in” to that client software.
- Everything we can do with HTML 4.01 tags, their permitted attributes, comments, and character entities, is defined in one of the HTML 4.01 DTDs.
- HTML5 is not really based on SGML and it does not need to be validated.

THE HYPERTEXT MARKUP LANGUAGE

When Tim Berners-Lee developed the WWW at CERN, he needed a document format that met a number of specific criteria. He chose to develop a text-based standard derived from SGML, where markup controls the document structure, its presentation, and the linking that adds the power of hypertext.

The HTML standard he developed is documented and maintained by the W3C (<http://www.w3c.org>). The HTML standard continues to evolve, and different versions and DTDs exist. HTML5, which is discussed later in this chapter, does not need validation against a DTD. Web documents might include various nonstandard proprietary tags added over the years by both Microsoft and Netscape during the browser wars of the 1990s. HTML files represent a subset of the world of text files, that is, they are text files where the format control codes are themselves simple ASCII text. HTML documents are therefore simply ASCII text files that have been “marked up” with special tags that control how the document appears when viewed by certain programs. Basing HTML on the ASCII text standard enhances the simplicity and portability of these documents. It also meant that no special programs were required to create the documents because almost all computer systems have access to a text editor, which, along with knowledge of HTML, is all one needs to begin creating Web content.

In HTML, certain text characters are reserved for the control functions that identify instructions. These instructions are called tags, and they are themselves simple text. Tags are identified by the less than (<) and greater than (>) symbols. Note that this is fundamentally different from most proprietary approaches to document structure and presentation. Because HTML reserves some characters that might also occasionally need to appear as text content, the DTD defines character entities for this purpose. For instance, if the < (less than) symbol is needed as content and not a tag identifier, a special name is referenced for it (a character entity). HTML uses tags to specify both the formatting and the logical organization of a document.

While the creation and modification of basic HTML files can be accomplished in any text editor, many programs exist that can format content as HTML. Word processing programs and dedicated Web-authoring programs provide a WYSIWYG view that does not require knowledge of the underlying source code. However, knowledge of source code is required for many kinds of troubleshooting and for certain tasks, even in user-friendly WYSIWYG-authoring programs. Because learning this language is an important objective, you should use HTML-authoring programs that permit direct interaction with source code. Many HTML editors have built-in tools to facilitate the insertion of appropriate HTML code, so it is not necessary to manually type in tags. Examples of such editing software are Chami's HTML Kit (<http://www.chami.com/>), SeaMonkey (<http://www.seamonkey-project.org/>), and NVU (<http://www.nvu.com/>), all of which have versions available for download and use.

There are ample examples of tags and coding provided in this overview of HTML, but experience is the best teacher, and as in learning any language, practice is required to become truly proficient. Some excellent websites with interactive tutorials allowing hands-on practice with HTML are listed in the resources section of this chapter.

The HTML 4.01 Specification

The HTML specification is an established standard maintained by the W3C. The newer HTML5 is designed for backward compatibility with HTML 4.01, so

it shares many of the same rules and properties described here. A few of the main items in the HTML 4.01 specification are:

- The document type declaration: This statement, with the version, language, and the DTD specification comes at the very beginning of every document. However, it is not actually part of the HTML document itself because it occurs outside of the root element (i.e., the HTML tag pair). HTML5 is not based on SGML and does not need a DTD reference; instead it uses the simple <!DOCTYPE html> declaration (W3C, 2011).
- Comments: Markup languages and scripting always allow for comment notes; an example is <!--this is a comment to you-->. The string <! introduces the *comment declaration*, and it can contain more than one comment. The comment itself is surrounded by double dashes.
- Character entities: These are special names or ASCII code number references associated with characters that may occur as content but are also reserved to identify markup codes (such as the “less than” symbol) or that require special display (such as diacritics, the tilde, accent marks). Character entities are referenced with the ampersand character (&) followed by either the named entity (such as “gt” for the > symbol) or a # combined with an ASCII character reference number. Both forms are closed with a semicolon (;) character.
- Elements and attributes: Elements in HTML are defined by tags. Attributes refine tags and they are expressed as a property with an assigned value, such as a source file designation in an image tag (**src = “somefile.ext”**).
- Empty elements: These elements do not markup any content. Most tags are containers of textual content, but some simply designate the insertion of an object such as an image or a horizontal rule.

In addition to defining the tags, the DTD also dictates the nesting rules. Figure 8.1 shows the basic structure of an empty HTML container (i.e., one with just markup and no content).

The first line, which begins with <!DOCTYPE . . . >, is really outside of the HTML container; it is the Document Type Declaration (not to be confused with

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Our Web page</title>
  </head>
  <body>
  </body>
</html>
```

Figure 8.1 A basic HTML container.

the Document Type Definition). It provides information about whether a DTD is used, and if so, which of the three HTML DTDs are associated with this instance document. Browsers do not formally validate documents, but they use DOCTYPE declarations to get information about what “mode” to assume in rendering the document. This doctype “sniffing” determines whether the browser renders the page in standards mode or the more backward-compatible “quirks” mode, which avoids “breaking” older, non-standard pages (Sivonen, 2013). This sample document was created according to the rules of HTML version 4.01 as referenced in the Transitional DTD maintained by the W3C consortium. As mentioned earlier, there are other HTML DTD versions: the Strict DTD and the Frameset DTD. The **<html> </html>** tag pair serves as the container for the entire document; because all other elements must be within this tag pair, it is referred to in the DTD as the *root element*. Like a directory tree, there is a hierarchy within the HTML structure: elements can have parent, sibling, and child elements. The root element (in this case the HTML tag pair) by definition is the only element that has no parent (just as the “root” directory has no parent directory). The HTML document has two main structural parts: the **<head>** area, which is reserved for metadata elements (such as the title tag shown), and scripting and style functions, discussed later. The **<body>** area is where all the document content must go. Note that nothing should be placed in the “no man’s land” between the end of the head area and the start of the body. The body area is where all the remaining nested tag pairs that mark up the content appear. By definition, in HTML anything that is not markup is content.

The structured nature of HTML also dictates the rules that apply to nesting order. Tags cannot overlap each other; they must be completely contained within other tag pairs. In addition, some tag nesting does not make sense. For instance, a heading level two should not occur within a heading level one; although it is not a nesting violation, the following example is not logical:

```
<h1><h2>This tag nesting is not logical</h2></h1>
```

An example of inappropriate tag overlapping would be the following markup, which attempts to present content as a heading level two with italicized text:

```
<i><h2>This is inappropriate nesting; the tags overlap.</i></h2>
```

If the italic tag is used with a heading, it should occur within the heading level two tag pair.

PHYSICAL VS. LOGICAL MARKUP

Even before the Web, documents formatted with word processing programs gave us the choice between controlling documents at the physical level (a procedural approach) or at a logical (or descriptive) level. In addition to the many font types and sizes that exist for text, fonts can have other properties such as being in bold or italics. Different structural parts of a document, such as a chapter heading or paragraph text, are visually distinguished from each other by applying different text decoration features to each. These properties must be controlled for each separate occurrence of these different forms of text

throughout the document. However, this approach has a number of limitations and inefficiencies when compared to the descriptive approach of defining and applying styles. Not only is procedural control more labor intensive, but style inconsistencies are more likely, and style changes are more difficult to implement. It is more efficient to designate what the text is structurally and then let the program handle the application of an appropriate style formatting for that class of text.

The same is true when controlling documents in HTML; the markup employed can be either procedural or descriptive. HTML has both forms of markup, also referred to as *physical* vs. *logical* markup. Physical markup attempts to control the physical appearance of a document, such as inserting the code that causes specific text to appear in italics. Another approach would be *logical* markup, which uses a tag such as the emphasis tag pair ` ` that informs the browser that the text is to be emphasized, but leaves it to the displaying software to determine exactly how it should appear. The markup options that favor logical over physical markup is another distinction of the Strict DTD as opposed to the looser Transitional DTD.

ELEMENTS, TAGS, AND ATTRIBUTES

Elements are parts of the logical structure of a document, including headers, paragraphs, tables, lists, and images. In HTML, tags mark an element and identify it to the browser; they identify document objects and, in the absence of other style rules, control presentation. Parsing programs recognize tags by the surrounding less than (`<`) and greater than (`>`) signs, which delimit them. Generally, HTML tags always occur in pairs with a starting tag and ending tag. Empty elements that do not always require an ending tag are discussed later. The ending tag is distinguished from the starting tag by the presence of a forward slash in front of the tag text. Document content appears between the beginning and ending tags; the tag pair can be thought of as a container of the content. An example of a tag pair that marks up heading level 1 content is:

```
<h1>This is a header</h1>
```

Tags identify elements and may have *attributes*. An attribute is a property, characteristic, or modifier of the element. Attributes are expected for some elements, but more often, they are optional. When an attribute is used, a *value* for it must be provided, and the value should be set within quotation marks. One important attribute used in HTML is NAME, which permits the element to be associated with an identifier. Note that in XHTML, the “ID” attribute replaces the use of “NAME.” Throughout this discussion, the older HTML convention of using NAME for an element identifier is used, and these are functionally equivalent. In later discussions of CSS and XML, the newer approach of using ID for NAME is used.

The previous heading example could be modified by adding the align attribute as shown here:

```
<h1 align = "center">This is a header that is centered</h1>
```

HTML permits both tag and attribute names to be in either upper or lower-case (but this is not the case in XHTML, as discussed later). Although HTML allows for some exceptions, generally the value string for an attribute must appear within quotation marks. The exceptions have to do with whether the values are literal strings or name tokens. Name tokens are assigned when an attribute can only have a limited and defined set of permitted values; for instance, the align attribute only allows values of left, center, or right. These name tokens do not have to appear within quotation marks (although it is good practice to use them in HTML, and they are required with XHTML). When the attribute value is unpredictable and can be any literal string (a file name for instance), it must always appear within quotation marks.

HTML also defines certain *empty elements*; they are by definition elements that do not “markup” any content and are therefore empty. Examples include the `
` tag, which forces a line break; the `<hr />` tag, which results in a horizontal rule; and the `` tag, which points to some image file to be placed in the page. Note that these empty element examples are not marking up any content; they also do not appear to have traditional ending tags. However, more rigorous approaches to HTML markup demands end tags for every element, so these tags are typically ended by placing the forward slash character before the closing greater than sign (`>`) as shown in the preceding examples.

The *Document Object Model* (DOM) treats a document as a collection of objects that give a structured and uniform programmatic interface for HTML, and elements represent specific objects. The DOM model is discussed further Chapters 9, 10, and 12.

Deprecated Tags and Attributes

Some older HTML tags or attributes are designated as *deprecated*; that is, although they are still supported by most browsers, they are being phased out because there are better ways to accomplish what the tag or attribute was intended to do. Examples of deprecated tags and attributes are the center, basefont, and underline tags. These tags have been mostly supplanted by the use of style sheets or simply dropped in HTML5. In addition, there are tags that are not deprecated, but whose use in certain contexts is discouraged; for example, using the table tag to control page layout as opposed to presenting tabular data.

Inline and Block-Level Elements

HTML specifies both *inline* and *block-level* elements. Inline elements are ones that appear without creating a new line break. Block-level elements always appear on a new line when displayed. Inline tags are rendered on the same line by a browser, and they can contain only text or other inline elements. Examples include the anchor, image, and font tags. Block-level elements can contain either inline elements or other block-level ones. This category includes tags such as those for heading levels, forms, tables, lists, and the frequently used division container. However, the browser default display of these elements can be

overridden with style directives; for instance, list items that are normally presented as block-level can be forced to display as inline elements if desired. The DIV tag is the generic block-level container, which is frequently used as a way to identify a named section of a document that can then be referenced in style rules. HTML5 replaces much of the need for generic **DIV** containers with new semantic tags to support specific container needs. In addition, HTML5 replaces the inline and block-level properties with a newer model. These differences are explored further in the section on HTML5.

USING HTML: COMMON TAGS

As with learning any language, the necessary starting point is to learn basic vocabulary and grammar; so to learn HTML, you must begin with some basic tags and their uses. A partial list of tags needed to start exploring HTML follows. There are many excellent resources on HTML, so recreating a complete HTML reference will not be attempted. (See the list of HTML resources for more information on the use of these and other tags.)

<html> </html> The root element that identifies an HTML document container.

<head> </head> The tag that identifies the head area; generally contains metadata as well as style and scripting functions.

<title> </title> The title tag within the head area that represents the minimum requirement for metadata.

<meta> Meta tags, also in the head area, contain generic or structured metadata.

<body> </body> Identifies the content area of the document.

<h1> </h1> A level one header, the structural element that indicates a section heading. Each subsequent level in the hierarchy is given a numeric value (h2, h3, etc.).

<p> </p> The tag that indicates a paragraph container. The ending tag is not strictly required except in XHTML.

 The tag that creates a line break (an empty element).

<div> </div> A generic block level container.

 A generic inline container.

 An image tag; an empty element that references an image to be displayed in the document.

<a> The anchor tag; the tag needed to create anchored hypertext references and as named anchors.

<table> </table> The table container; tables contain <tr> </tr> tags for rows and rows contain <td> </td> for table data.

 An unordered list.

 An ordered list.

 A list item.

<dl> </dl> Definition list that contains the tags <dt> </dt> definition term and <dd> </dd> for definition.

Preformatted text: <pre> </pre> Tells the browser to maintain the text layout as it would appear in a text editor view.

<hr /> A horizontal rule that creates a lined division across the page.

Images and HTML

HTML files are just a special type of simple text file. How can a text format accommodate a binary image object? The simple answer is that it cannot; it can just point to where such an image resides on a server and it must be sent to the client in a subsequent transaction. The markup informs the browser to display the separate image file at the specified location in the document. This is fundamentally different from the way word processing programs incorporate images, which embed them as an object within the document proprietary binary format. When an acceptable image type is placed in such a document, that binary object becomes a part of the code that makes up the document file itself, resulting in a single file container that has both the text and the image available to the displaying program. This is very efficient but also requires that those using the file have the same word processing program to view it.

HTML is a text format and therefore can only use a text directive (the image tag and source attribute) that points to some external image file that resides outside of the HTML file container on the server. When the browser requests an HTML file, the server sends that file. The browser then must make separate HTTP requests for the image files referenced in the image tags. The image tag is one of a small set of empty elements (no content is “marked up” by it). The image tag has a required attribute for the source file and another expected attribute for alternate text, which is useful for all displaying programs and essential for accessibility compliance, as in the following example:

The source attribute is shown with the **src = “string,”** which specifies the name of the required image file. If the image file is not in the same directory location as the HTML file, the path to the appropriate location must be specified along with the filename. The **alt = “string”** gives the alternate text that displays when the cursor hovers over the image in most browser views. Attribute values should appear in quotation marks and, if that value is a filename, it must be an exact match with the actual name. This is complicated further by the use of mixed case for filenames. The hosting platform is often a UNIX host, which is a case sensitive OS. Case sensitivity is a source of potential problems because Windows is not case sensitive and tends to apply uppercase letters for file extensions with many programs. Most authoring programs let you add an image to HTML by “browsing” to the image on a local drive and selecting it. When the source code is actually examined, the filename may be in upper or mixed case, which means that often the source code must be edited or the referenced file renamed to ensure the HTML reference and actual name match exactly. For this reason, it is a good practice to enforce a consistent naming

convention for all files and directories created for a website. Sometimes browsing to an image file in an editing program will also lead to the inclusion of a local drive and directory path to the image reference in the HTML. This should be checked because local path references will not match the path needed on the hosting site once the HTML file is uploaded. Image tags also employ attributes to specify the height and width. However, these must be altered proportionally to avoid distortion.

The Hypertext Reference

Much of the power of the Web is due to its hypertext capabilities, and the successful creation of meaningful hypertext references is an essential role of the Web publisher. The anchor tag has several important attributes that alter its function, but when used to create the standard link, the hypertext reference (**Href**) attribute is specified. The browser default renders the content that appears between the beginning and ending anchor tag as the blue underlined text universally recognized as a link. Style applications can change this default formatting, but the conventional blue underlined style is so ingrained that users often do not recognize alternate link formats as links. An example of an anchored hypertext reference is shown here:

```
<a href = "minutes.htm">Meeting minutes</a>
```

The browser default is to display the link text “Meeting minutes” in blue with underlining, and when the text is clicked, the file “**minutes.htm**” is requested from the server. From the information in the HREF value string, the file “**minutes.htm**” must be in the same directory as the file that contains this source code because no other path information is provided. The various ways to make these references are discussed in the section on absolute vs. relative references.

In addition to simply making a link to a file, which loads the file starting at the beginning of the file, it is sometimes desirable to point to a specific place in that file. To do this, a type of “bookmark” is created in that file that can then be referenced in the link HREF string. Assume that the “**minutes.htm**” file is quite long and that you want a reference that jumps to a specific section of the file that begins with a heading called “Proposals.” The first step is to create a named anchor in the file “minutes.htm” at the appropriate place, as shown here:

```
<a name = "propose"><h2>Proposals</h2></a>
```

To do this, the anchor tag attribute “**NAME**” is specified instead of the “**HREF**” attribute. Note that the name attribute is used in this example but it is deprecated in XHTML and not supported in HTML5, where the “**id**” attribute is used instead. In addition, although the anchor tag pair shown encloses the H2 tag pair, it does not have to “contain” any text because the tag just acts as an “invisible” placeholder that can be referenced later in a link. Finally, note that the string used as the value for the “**NAME**” attribute can be anything; the name could be “xyz” or just a number. Any anchor name is permitted as long as it does not have any spaces in the string and it is referenced correctly in the

link that will use it. Once a named anchor is created, a link can be made to that particular place in the file, as opposed to simply displaying the document from the beginning. To do so, the link needs to reflect not just the filename but also the named anchor within that file, as shown here:

```
<a href = "minutes.htm#propose">Meeting minutes—proposals</a>
```

The pound sign (#) separates the filename from the named anchor. This technique can create links to other places within the same file providing quick access to various parts of a long document by beginning with a table of contents with links to various document sections. In the case where the link and the named anchor are in the same file, the **HREF** value does not need to include the filename, as shown in the following code:

```
<a href = "#propose">Proposals</a>
```

This named anchor is assumed to be in the same file container as the file that contains this link because the HREF contains no filename.

Relative vs. Absolute References

Whether you are pointing to some image source file or making a link with the anchored hypertext reference, it is critical that the location of the external file is accurately reflected in the reference. There are two ways to specify any external file. The reference can be *absolute*, which provides a complete URL that unambiguously defines where the resource is on the Web. Alternatively, the reference can be *relative*, which provides directions as to its location in relation to the location of the HTML file that is pointing to it. When an external resource is on the same server, it is very common to use relative references. This might be as simple as providing the name of a file located in the same directory. However, because it is a common practice to create subdirectories (i.e., folders) to organize the files that comprise the site and co-locate similar types of content or formats, it is useful to know how to reference other locations in the directory tree. Making most references relative simplifies the source code, facilitates testing of links locally, and makes site maintenance easier and more efficient. For instance, a server administrator may relocate a group of folders to another place in the directory tree, but as long as the relationship among the adjacent directories stays the same, the links and image references still work after the move. Another way to handle this issue of directory tree relocation is with the use of the **<base>** tag, which allows for a base URL to be set that is independent of the referring page location. This is possible because a server never actually sees relative URLs. As described in the discussion of HTTP, when a user selects a link constructed in a relative fashion, the server constructs the full URL path by appending the location information of the referring page to the truncated URL in the browser request header.

A conceptual model or map of the hierarchical directory tree structure is required in order to make valid relative references. Many Web servers are UNIX based, and these systems have elaborate directory structures. The root directory

is the starting point on such a system and is the one directory that by definition has no parent directory. Directories below the root are “level one” directories and have the root as the parent; subdirectories below a level-one directory are level-two directories, and so on. All directories then (except the root) have a parent that is one level above them in the tree; subdirectories are referred to as child directories and would be one level further down in the tree; and directories at the same level in the tree share a common parent and are therefore sibling or parallel directories.

In addition to the machine tree structure, the Web server program is configured to establish a type of “virtual root” that the program uses when seeking Web documents; this location is typically named **HTDOCS**. Unless directed otherwise, the Web server looks “down” in the tree from some current location, which is defined simply as the directory that contains the HTML file with the link code. Therefore, to reference a file in a child directory, the path requires just the name of the child directory along with the filename. To reference a parent directory, the “two dot (..)” convention is used; a “single dot (.)” explicitly references the current directory. A hypothetical directory tree is shown in Figure 8.2.

In this directory tree, assume that a file called `welcome.html` is stored in the directory called `school`, which is a subdirectory of **htdocs**, which is a subdirectory of **www**, which is below the root. The directory called **graphics** is a parallel (sibling) directory to the directory **school**. If there is an image called “**banner.gif**” in a Web page called “**welcome.html**” that is stored in the directory called **school**, there are two ways to make the source reference in the image tag: an absolute reference or a relative reference.

First, consider the construction of an absolute reference. As noted earlier, Web servers are set up to start the search path to Web resources at some location other than the true system “root” directory. Assuming that the location **htdocs** in Figure 8.2 acts as a “virtual root” for the server, the absolute reference would be, ``

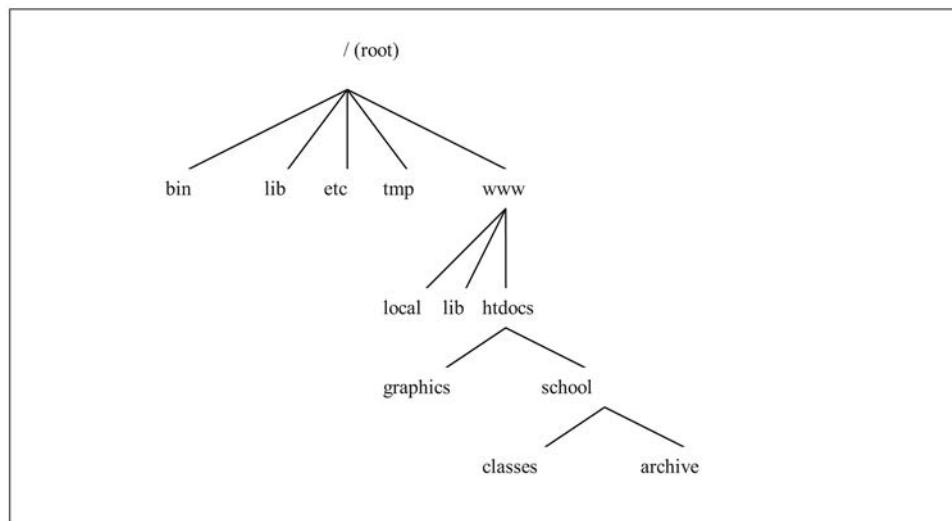


Figure 8.2 A hypothetical directory tree.

Alternatively, this could be done as a relative reference. This approach simply provides the path that describes where the image file is in relation to the location of the referring HTML file. Because “**welcome.html**” is in a sibling directory of the location called **graphics** that contains the image file, the path to the image must first specify a directive to go up one level in the tree (to the parent directory) using the dot-dot (..) convention. From there, the path continues down into the **graphics** location where the requested image file is located. Relative references could be thought of as similar to a “mall map” that shows how to get to other stores from your current location. Such a map visually places you in a current location with a “You are here” marker and then shows how to get to other locations from that place. On the server, the “You are here” location is the current directory where the file with the HTML code is stored. The relative reference for the image would be, for example, <**img src = “./graphics/banner.gif”**>.

The same idea applies to anchored hypertext references. Assume a link is needed from “**welcome.html**” to a file called “**LIS600.html**,” which is stored in the directory called “**classes**” in Figure 8.2. Recall that the “**welcome.html**” file is in the directory **school**, the parent location of the **classes** directory. Again, there are two ways to make this reference. As before, an absolute reference could be made. Remember that the location of **htdocs** is like a “virtual root” for the Web server and is not explicitly specified in the path. The full URL in the absolute reference completely and unambiguously defines this file as a point of content on the Web in this link tag, as shown here:

```
<a href = “http://servername.com/school/classes/LIS600.html”>
```

A relative reference could also be used instead. This simply supplies the path to locate the file from the location of the referring file. Because the location of “**LIS600.html**” is in a child directory of the current location, to get from where the referring file is to the *directory location of the resource referenced* the only required information is the child directory name along with the filename. This reference could include the explicit notation for the current directory (a single dot followed by a forward slash) at the beginning of the **HREF** string, but it is not required. The relative reference is shown here:

```
<a href = “classes/LIS600.html”>
```

The relative reference for this link is not only simpler, but it also still works if the **school** portion of the directory tree with these subfolders is relocated on the server. The ability to write relative and absolute references correctly is an extremely useful HTML-authoring skill. A few common technical errors consistently crop up in making these references, including:

1. Not using a consistent file-naming scheme and directory structure. Because many hosting servers are UNIX based, they are case sensitive. Names such as “**Banner.GIF**” and “**banner.gif**” are treated as completely different names.

2. Confusing the two slash angle variations. Windows use a backslash (\) to designate the root and to separate directory and filenames from each other. UNIX-based systems use a forward slash (/).
3. Failure to specify protocols in URLs. For absolute URLs, HTML code requires that the protocol be included. A HREF value of “www.uky.edu” needs the beginning protocol designator, as in “http://www.uky.edu.”

Tables

Tables are useful for presenting data, but they have been widely abused in HTML. In addition to simply presenting tabular data, borderless tables without visible gridlines were often employed as a strategy to control layout and formatting of HTML documents. This use of tables is discouraged because CSS is the recommended way to control presentation; in addition, this use of tables causes accessibility problems as well. However, as mentioned in the discussion of mobile design, they are sometimes used this way as part of a design solution. As in word processing, HTML tables are used for presenting tabular data. The markup for tables is straightforward but detailed because tables require that each row and data cell be specified. A simple HTML table container is in Figure 8.3.

```
<table summary="">
<tr>
    <td></td>
</tr>
</table>
```

Figure 8.3 HTML code for a simple table.

Tables may have a border surrounding them or no border, depending on the border attribute. Tables may cause problems for display programs, especially with screen reading programs used by the visually disabled, but the inclusion of the summary attribute at least provides descriptive information about

the tabular content to those programs. Table headers use the **<th> </th>** tag pair. Table rows are created with the **<tr> </tr>** tag pair, and table columns are determined by the creation of table data cells with the **<td> </td>** tag. The colspan and rowspan attributes allow the spanning of columns and rows as needed. Finally, cell spacing and cell padding controls can be added as table attributes to modify the appearance of the table.

Forms

HTML forms have many important applications on the Web. They are used to create the interface for search queries with engines and library databases as well as deliver user feedback or questions. They are commonly used for reference questions, surveys, interlibrary loan requests, and orders. The use of feedback forms is a more user-friendly method of soliciting users' questions or comments compared to simple “mailto” links that depend on the availability

of a local mail client program. **mailto** links are created with an anchor tag where the HREF value uses “mailto” in place of a protocol assignment and an email address in place of a URL. However, **mailto** links often do not work as expected because they depend on the client computer configuration and the mail software of the end user. For instance, even if a mail client program has not been setup on the user’s machine, a “mailto” link nevertheless often launches a mail client such as Outlook Express anyway, leaving users confused, or worse, giving them the impression that a message has been sent when it really might be just queued up in the outbox and never actually sent. The proper use of HTML forms also requires an understanding of how data is posted to server-side scripts; this aspect of form use comes up in the discussion of HTTP protocol methods. There are options for adding forms that do not require knowledge of HTML form coding that will embed forms within a web-page. For example, Google Drive has a form creation option that will provide the HTML code to easily embed the form into a page. An added advantage is that this Google form will also capture the data in a Google spreadsheet. An example of the HTML coding to accomplish this type of embedded form will be examined later.

The tag coding needed to begin a form, not surprisingly, is the **<form> </form>** tag pair. The form tag has two essential attributes: **action** and **method**. Generally, forms send data to a script identified in the value of the action attribute. The method in the form is determined by how the script was written; often the script expects the HTTP method *post*. The input tag determines the options for data entry sent to the script. Some commonly used form tags:

- **<form action = “some_script” method = “some_method”> </form>**
Identifies a form container along with the script it interacts with and the method for the interaction.
- **<input> </input>** Form input elements usually always have the **type** and **name** attributes defined. Sometimes a specific data *value* is also specified if the value to be sent to the script is predefined.
- **<select> </select>** Creates a select list of options to choose from.
- **<textarea> </textarea>** Creates a textbox where comments can be placed.

Figure 8.4 shows the HTML code for a simple form; Figure 8.5 shows how a browser would render that code.

The form code example begins with a form tag to identify the form container, as shown here:

```
<form action = “http://www.uky.edu/AnyFormTurbo/AnyForm.
php” method = “POST”>
```

The action attribute specifies a php script called Any Form that resides in a specific location on a Web server. This particular script has a relatively simple job: to accept the data from the form of variable name/value pairs in the script and use them to create and send an email message. (Note: this script does not accept requests from outside this network, so if you want to try this out, you

```

<html>
<head><title>SLIS Info Request Form</title></head>
<body>
<h1>Request for Class Information</h1>
Please use the following form to request an information packet.
<br>

<form action="http://www.uky.edu/AnyFormTurbo/AnyForm.php" method="POST">

<input type="hidden" name="AnyFormMode" value="mail" />
<input type="hidden" name="AnyFormDisplay" value="Standard" />
<input type="hidden" name="AnyFormTo" value="jbmiller@uky.edu" />
<input type="hidden" name="AnyFormFrom" value="class Web page form" />
<input type="hidden" name="AnyFormSubject" value="Class Information" />

<p>Place your cursor in the first box, then use TAB to move to the next box. </p>
Please enter your name: <input type="text" name="nm" size="30"><br />
Your Street Address: <input type="text" name="street" size="30" /> Apt # <input type="text" name="apt" size="4" /><br />
Your City: <input type="text" name="city" size="20" /> State <input type="text" name="st" size="2" /> Zipcode: <input type="text" name="zip" size="5" /><br />
<br />
Please rate your computing skills:
<Select name="Self Rating">
<option>Newbie</option>
<option>Middle</option>
<option>Expert</option>
</select>
<br />
Are you Male <input type="radio" name="gender" value="male" /> or
Female <input type="radio" name="gender" value="female" />
<br />
Your email address if you have one:
<input type="text" name="email" size="30" />
<br />
Enter any comments or questions here:<br />
<textarea name="comment" rows=6 cols=40></textarea>
<p>
Thank you for your interest.
</p>
<input type="submit" value="Send Info" /> to the instructor.
<input type="reset" />
</form>
<hr />
</body>
</html>

```

Figure 8.4 HTML code for a form.

will need to locate or create a different script!) The following tags are types of input tags:

```

<input type = "hidden" name = "AnyFormMode" value = "mail" />
<input type = "hidden" name = "AnyFormDisplay" value = "Standard" />
<input type = "hidden" name = "AnyFormTo" value = "jbmiller@uky.edu" />
<input type = "hidden" name = "AnyFormFrom" value = "class Web page form" />
<input type = "hidden" name = "AnyFormSubject" value = "Class Information" />

```

The screenshot shows a web browser window with the title "SJS Info Request Form". The URL in the address bar is "sweb.uky.edu/~jbmill00/joe-form.htm". The main content is a form titled "Request for Class Information". It contains the following instructions: "Please use the following form to request an information packet. Place your cursor in the first box, then use TAB to move to the next box." Below this are several input fields:

- "Please enter your name:
- "Your Street Address: Apt #
- "Your City: State Zipcode:

There are also dropdown and radio button fields:

- "Please rate your computing skills:
- "Are you Male or Female
- "Your email address if you have one:

A large text area labeled "Enter any comments or questions here:" is followed by a "Send Info" button and a "Reset" button.

Figure 8.5 A browser view of the form code.

As the “input” label implies, these tags accept input to send to the script. Note that they are also “empty elements” that do not markup any content. They do not require an end tag, but it is good practice (and an XHTML mandate) that they all have an ending forward slash before the greater than symbol to serve as the end tag notation. These input tags have three defined attributes:

- The type = “hidden” attribute means that these tags do not result in any display in the browser; they are used as a way to pass data to the script without interaction with the user.
- The name = “string” attribute creates a variable name that is associated with some value.
- The value = “some_value” attribute pre-assigns the desired value to the variable name.

Hidden input tags send the variable name/value pairs that the script uses to format and address the email message that is created when the form is used. A variety of input tags intermixed with descriptive content and formatting HTML follows:

**Please use the following form to request an information packet.
Place your cursor in the first box and use TAB to move to the next
box.
**

Please enter your name: <input type = "text" name = "nm" size = "30" />

Your Street Address: <input type = "text" name = "street" size = "30" /> Apt # <input type = "text" name = "apt" size = "4" />

Your City: <input type = "text" name = "city" size = "20" /> State <input type = "text" name = "st" size = "2" /> Zipcode: <input type = "text" name = "zip" size = "5" />

The section above has six input tags of **type = "text."** These tags result in text boxes of the size specified in the size attribute. The name attribute creates four new variable names, each of which is associated with whatever data the user puts into the text box displayed in the browser window. The next section creates a select list:

```
<Select name = "Self Rating">
<option>Newbie</option>
<option>Middle</option>
<option>expert</option>
</select>
```

The select tag results in a new variable name called “self rating” and a click down list of options. The option tags create the available choices. The next section creates a variable called “gender” and displays radio button choices:

```
Are you Male <input type = "radio" name = "gender" value = "male" />
or
Female <input type = "radio" name = "gender" value = "female" />
<br />
```

A value is preassigned with each choice in these **type = "radio"** tags; because the same variable name is used, the choices become mutually exclusive (i.e., only one choice is permitted; selecting one radio button deselects the other). Next is another **input type = "text"** and an open-ended text area for comments:

Your email address if you have one:

```
<input type = "text" name = "email" size = "30">
<br />
```

Enter any comments or questions here:


```
<textarea name = "comment" rows = "6" cols = "40"></textarea>
<p>
```

Field Name	Entered Value
name	Mary Smith
street	19 Oak Ave.
apt	
city	Lexington
st	KY
zip	40506
Self_Rating	Newbie
gender	female
email	mary.smith@email.com
comment	I am interested in this class

The Above Data Has Been Sent.
AnyFormRandomSeqNo: 95638112

Figure 8.6 The variable name/value pairs sent to the script when a hypothetical student fills in and sends the form.

The final input tags in the form result in the creation of buttons for “submit” and “reset,” followed by the ending form tag, shown here:

```
<input type = "submit" value = "Send Info" /> to the instructor.
<input type = "reset" />
</form>
```

If student Mary Smith filled in this form, as shown, the variable name/value pairs shown on the left in Figure 8.6 are sent to the script, which are sent to the email address provided in the Any Form To values.

Image Maps

Image maps, also referred to as “clickable images,” provide a mechanism to associate areas of an image with different URLs. Separate from mapping, any entire image can be made into a single link simply by enclosing the image tag within an anchored hypertext tag pair, as opposed to using some string of text for the link. However, it is sometimes desirable to make a series of “hotspots” within a single image that enable more than one hypertext reference to be associated with it; such an image can be useful as a navigational graphic or site map image. There are ways to accomplish this effect with scripting, but image maps are still an option. The image area to be associated with a URL is defined in terms of pixel areas within an X-Y coordinate scheme. The X-axis is the horizontal grid line and starts with pixel 0 at the extreme left; the Y-axis

is the vertical grid line, starting with pixel 0 at the top. A map file or map tag container holds the coordinates that define the shape and associate the link with the defined image area. Originally, image maps were all *server-side*; that is, the map file had to be stored and processed on the server. However, browser clients can now process these instructions, so it is more common for the map instructions to be embedded in the HTML code. The client in the later case does all the processing; hence the name *client-side* image maps. Although you could create such a set of map instructions manually, there are specialized programs that facilitate this process; in addition, this functionality is often available in various Web authoring and graphics programs.

Both server-side and client-side approaches require that an image be available with which URL hotspots can be associated. One of two different attributes of the image tag identifies the image map type. Server-side mapping is associated with the **ismap** attribute, which means the image is an active image. This requires the image tag be enclosed within an anchor tag that specifies the URL of the server map file. The cursor location data is then sent to the server when the hotspot is clicked on. For example, the link code below identifies a server-side map:

```
<a href = “/htbin/htimage/search.map”><img src = “graphics/
search.gif” ismap></a>
```

The client-side approach uses the **usemap** attribute in the image tag. The name value for the **usemap** attribute must match the name associated with the map tag container. The map tag instructions are usually located in the same HTML file as the image tag. With the client-side map, the coordinates are specified in the HTML source code and interpreted by the browser. The code below identifies a client-side map, and the “#king” indicates there must be a **<map>** tag container with a matching NAME attribute in this HTML file that associates image coordinates with links:

```
<img src = “king.gif” border = “0” usemap = “#king”>
```

Framesets and Frames

Framesets were an early design strategy that allowed the screen to be divided into different windows that were filled with separate files. A common use of frameset designs was to provide an index file of site links that remained visible on the left side of the screen while displaying the selected document content on the right. Although the frameset tag is not officially deprecated in HTML 4.01, they are nonetheless considered obsolete and they are not supported in HTML5. The use of framesets fell out of favor for a number of reasons. One issue is the way search engines find and index the files and content; when framesets are used search results often led users to content outside their intended context. There were also issues about how links displayed in framesets and they created serious accessibility problems. CSS provides a better alternative to create this layout effect. However, there are still occasions where framesets have been employed or where you may have to deal with their use whether they are a good design choice or not; for instance, the some course management systems

display content within a frame environment. Separate from design considerations, an examination of how to control where links display between frames demonstrates several useful tags and attributes. Although framesets are not recommended, a variation of the frame idea is still useful. The `iframe` tag is often used to embed videos, forms, or other page components.

Framesets are codified in HTML 4.01 Frameset DTD. Framesets consist of multiple files that work together to present the screen layout and content; they use a master frameset file that acts as a screen blueprint and required additional HTML files to fill each frame window. Frame tag source attributes identify the files intended to fill each window created. Other frame tag attributes include controls for appearance and presentation. The “`noresize`” option determines whether the user can resize the frame window in the browser, and scrolling options can be turned on or off. The “`name`” (or also “`ID`”) attribute assigns each window a unique identifier, which, when referenced as the target for links, allows the page to load in a window other than the window that contains the link.

Inline Frames

A common use of the frame concept now is the **inline frame** or **floating frame**, which uses the `<iframe> </iframe>` tag pair. The element defines an inline frame window for the inclusion of external objects such as embedded videos and can function as a target window in which linked content can appear. It is similar to the object element, which also allows for the insertion of a HTML document within another; inline frames are another way to nest a document within a document. The `<iframe>` element has many of the same attributes

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>An inline frame</title>
</head>
<body>
<h1>A page with an Inline Frame</h1>
This page has an inline frame. For instance, it could be used to display additional explanatory text when highlighted terms are selected.

<iframe src="sidebar.html" width="200" height="300" scrolling="auto"
frameborder="1" align="right">
[Your user agent does not support frames or is currently configured
not to display frames. However, you may visit
<A href="sidebar.html">the related document.</A>]
</iframe>

</body>
</html>
```

Figure 8.7 HTML code that creates a floating frame.

described for the frame tag. A source attribute (**src**) informs the client what file should be displayed in the object window, the **name** (or **id**) attribute identifies the window as a target location in which linked content can appear, and the standard width, height, align, scrolling, frameborder, margin width, and margin height attributes control the frame window appearance. Figure 8.7 shows the code for such an iframe that will display the content of the file “sidebar.html,” followed by its appearance in a browser display, shown in Figure 8.8. As mentioned earlier, forms can also be embedded in this way using services such as Google Drive. The embed code generated by Google for a form created in this way shown in Figure 8.9.

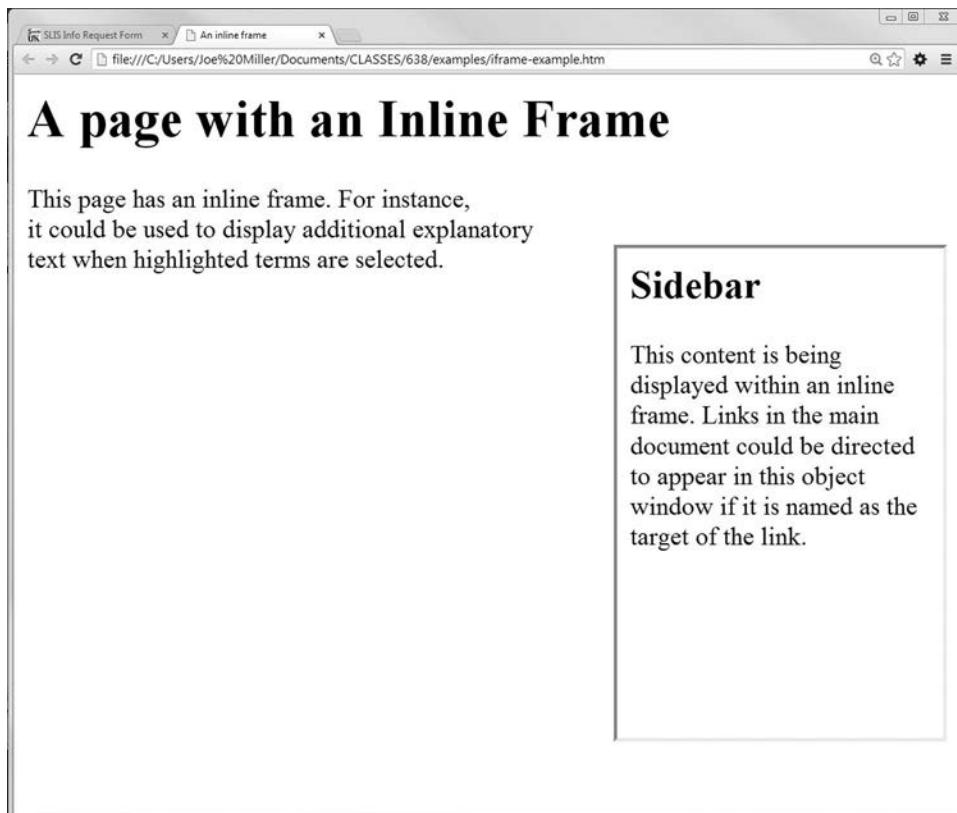


Figure 8.8 Browser rendering of the HTML code in Figure 8.7.

```
<iframe src=
"https://docs.google.com/forms/d/18g0e_tDZRB36d1_EH3y4VV5wCqrUi5OTalkLdBIK_EA/vie
wform?embedded=true" width="760" height="500" frameborder="0" marginheight="0"
marginwidth="0">Loading...
</iframe>
```

Figure 8.9 The iframe embed code for a form made on Google Drive.

Hypertext Links and Frames

The default display for a link within a frame window is for it to appear in the *same window* that contains the link unless there are instructions that direct the link to appear in a different window location. If the link is supposed to display in either a different frame or iframe window, several additional tag attributes are needed. First, a **name** or **id** attribute is used in the frame or iframe tag to assign a name value to it, for example: **<iframe src = "file.htm" name = "window1">**.

The specific choice of a value for the name attribute is unimportant; it simply serves as an identifier that can be referenced later in links. The name can then be used in links as a way to control where the linked content appears. To force a link to display in a particular iframe window, the **target** attribute is added to the anchored hypertext reference to identify the object window in which it should appear.

In addition to targets we create to match iframe or frame windows, there is a set of predefined targets, consisting of four “magic targets.” They all begin with an underscore character:

1. “_blank” launches the linked resource in a new window;
2. “_self” loads the resource in the same window in which the link appears;
3. “_top” loads the resource in the full body of the window;
4. “_parent” loads the resource in the immediate frameset parent.

The Base Tag

Another useful tag is the **<base />** tag. HTML allows the setting of a **base target** that controls the window location of all links unless overridden within a particular link. The base tag is placed in the head area of the index.html file. An example of its use is shown here, where the name value matches the name of a frame or iframe window: **<base target = "right" />**.

The base tag may also be used to create a base URL for a file with the **Href** attribute. In the earlier discussion of HTTP, a standard “Get” request header was shown to include the URL of the referring page. However, if a base tag specifies an HREF value different from the actual location of the file that contains it, that HREF location is substituted for the actual location for any relative links it contains. This use of the base tag can thus maintain the integrity of relative links if the referring page has been relocated to a different location in the server directory tree.

XHTML

XHTML is a reformulation of the HTML standard to comply with the rules of XML. HTML and the browsers that display it were designed to provide a

relatively forgiving environment. Many HTML documents were (and are) created by nonexperts, and consequently often have errors in the code, such as the absence of a root element, open tags, and various nesting errors. Browsers ignore most of these common mistakes, and errors do not stop the processing of the rest of the document. When the browser encounters these errors, it usually just displays what it can of the document. XML is much less forgiving of such errors; a parsing error results in the display of an error message with no other content. The reformulation or migration of HTML into XHTML is a goal of many Web publishers, and new content is usually expected to comply with XHTML. The specific requirements of XHTML that HTML does not strictly enforce are:

1. All documents must contain a root element.
2. All tags must be closed, even empty elements.
3. All tag and attribute names must be written in lower case.
4. All attribute values must be enclosed within quotation marks.
5. Tags must be properly nested.

Empty elements, defined as those that do not markup any content, require specific attention because most early HTML programs did not apply end tags to them. Examples of empty elements previously discussed that may commonly appear without end tags are the image tag ``, the line break `
`, and the horizontal rule `<hr>`. A forward slash placed at the end of the tag string ends these tags, for example, as in this image tag: ``. As mentioned earlier, the use of name tokens that are predefined values for some attributes did not require quotation marks in HTML, but because quotation marks are required for all values in XHTML, it is good practice to use them consistently. Another change with XHTML alluded to earlier is the use of the ID attribute to create an element or object identifier in place of the NAME attribute.

MOBILE MARKUP LANGUAGES

New markup languages have been created to address specific issues and developer needs for the mobile Web. Several of these have faded, such as WML, which was an initial attempt to overcome the limitations of traditional HTML for delivering content to mobile devices. Others, such as XHTML and HTML5 have broader application and significance.

WML

In the late 1990s, interest in mobile Web led to the creation of WML (Wireless Markup Language), a markup specification to accommodate the limitations of mobile devices. The first WML standard evolved from several early attempts to develop mobile standards and proprietary languages solutions from device makers Nokia and Ericsson. The WAP forum created WML 1.1 in 1998 and

it was revised in 2001, but it was not widely adopted. XHTML-MP, described next, was more successful.

XHTML-MP

XHTML 1.0 with CSS 2.1 was considered the “standard” for Web development, but mobile devices introduced unique challenges that required developers to think “outside the box” (which can be taken somewhat literally in the context of the CSS box model). In 2002, WAP 2.0 allowed mobile browsers to display basic XHTML without CSS support. This led to the development of XHTML-MP, a limited version of XHTML with special features to support mobile that does support some CSS; a variation called CSS-MP was developed to provide style support for XHTML-MP. XHTML-MP version 1.2 was often the best solution to support users with older mid-level mobile devices with browsers that are not HTML5 capable (Gardner & Grigsby, 2012). It did away with HTML frames, iframes, and link target attributes that made no sense with mobile screens. In addition, it supported access keys that allow links to be associated with keypad numbers 0–9, accomplished with a new anchor tag attribute. For example:

```
<a href = “#section1” accesskey = “1”> Go to section 1</a>
```

XHTML-MP 1.2 has been superseded by XHTML-Basic 1.1, which includes the entire MP version with a few added features. However, neither are always viable mobile solutions because not all devices/browsers do well with this approach. In addition to these XHTML options, some large mobile projects use XML and XSLT in conjunction with other markup languages such as WML or XHTML-MP. Additional discussion of CSS for mobile is found in Chapter 9 and at <http://www.w3.org/TR/css-mobile/>.

HTML5

HTML5 is viewed by many as another answer to the challenge of mobile Web development, but it has broader applications beyond just mobile because it represents a major shift in the markup language of the Web. A recommended source on HTML5 is that of Lawson and Sharp (2011) and a good overview is by Suetos (2010). Although it has not yet been widely deployed, this standard has a relatively long history. Throughout the 1990s, the HTML standard was maintained by the W3C and version 4.01 represented the latest version. By 1998, many thought the future of the Web was going to be about XML, and standards development focused on XHTML implementation as a transition to that world. XHTML transitional was to begin this shift followed by XHTML strict, which was mostly a “stricter” approach to HTML that enforced XML rules (those rules and XML are the focus of Chapter 12). However, XHTML 2.0 was a more radical change that created “backward compatibility” problems and many questioned this view of the future Web.

Against this backdrop, a number of groups began exploring another way to extend the functionality of HTML with what would become HTML5. This standard has not been the work of one individual or organization. It began with a group at Opera working on extending HTML forms while maintaining backward compatibility; this became Web Forms 2.0, which later was integrated into HTML5. The effort was joined by Mozilla programmers and collectively they formed the WHATWG (the Web Hypertext Application Technology Working Group) to continue HTML development (see <http://www.whatwg.org/> for more on that initiative). In 2006, W3C realized the Web would not jump in one leap to XML, and they joined the HTML5 effort, along with Google, Microsoft, and other major developers. Although these two groups (WHATWG and W3C) have not always agreed on specifics, there is a consensus on some common guiding philosophical principles. HTML5 is predicated on some core design ideas. As summarized by Lawson and Sharp (2011), the underlying goals for HTML5 should:

- Specify interoperable browser behaviors.
- Define error handling in a consistent way.
- Evolve the language to make Web applications easier to create.
- Maintain backward compatibility with HTML 4.01.

Maintaining backward compatibility is challenging because there is a good deal of “bad” HTML out there such as tag nesting errors, unclosed tags, and missing structural elements as well as older pages with legacy or proprietary tags. Different browsers handle these issues somewhat uniquely. HTML5 is defined in terms of DOM, which is discussed further in the chapter on Web programming. This is important because errors in HTML can create different DOMs in different browsers, which in turn can cause problems with CSS styles and JavaScript. The HTML5 specification allows for more consistent error handling with a single DOM telling the parser how to handle “bad” code. HTML5 continues to define legacy elements such as the `` tag; the HTML5 parser can handle all HTML.

Key Differences: HTML5 vs. HTML 4.01

As noted earlier in this chapter, HTML5 is not really based on SGML the way HTML 4.01 was and it does not need to be validated. HTML5 is not an XML language either; HTML5 is its own language. Although it is not XML, it does introduce semantic markup, making it a superset of HTML. These semantic elements were partially derived from an analysis of HTML class and DIV container names, which are discussed further in the next chapter on CSS. Recall that the DIV tag is a generic block-level container, and as explained in the next chapter, this container can be referenced later in CSS through a matching ID value. In HTML 4.01, generic DIV containers are often used with ID values associated with common document areas such as menu areas or sidebars. HTML5 builds many of these structural components into the markup itself with new elements

that utilize semantic tags for these common DIV containers, including header, footer, nav, section, article, aside, and article. In addition, HTML5 includes new tags for multimedia such as the <video> tag for incorporating video instead of using a generic <object> tag. This allows better control and interaction with CSS and JavaScript as well as reducing dependence on external plugins for multimedia display.

Another distinction is that the DOCTYPE declaration is simply <!DOCTYPE html> with no URL reference or version noted. HTML5 also uses a new content model. Whereas HTML 4.01 and earlier defined elements as block level or inline, HTML5 does away with these presentational defaults and uses new models in their place. These are *phrasing content*, which is similar to the way inline elements were handled, and *flow content*, which is similar to HTML 4 block-level elements.

Surprisingly, HTML5 is more lax than you might expect about some elements. It does not require structural elements such as <head>, <body>, or even <html> because they are assumed and most browsers would display the page without them unless XHTML syntax was enforced.

Many Web 2.0 sites often behave like apps for mapping, image editing, or word processing. HTML5 supports this type of Web application development—these apps depend on scripting, and HTML5 specifies a new DOM API for drag and drop that makes it easier to write reliable Web apps. This open standard provides developers with an alternative to proprietary app environments such as Flash.

Major Advances with HTML5

To summarize, major advantages of HTML5 include:

- Self-validating forms: new attributes that improve form validation.
- Better multimedia support: elements for video and audio with improved media controls.
- The <embed> tag is formalized in HTML5; because of differences in native browser video support, plugins are still sometimes needed for multimedia playback.
- It facilitates better client-side data storage by replacing cookies with APIs for cloud storage or through the creation of a client-side SQL database that the browser can access.
- It supports better offline caching for Web apps, which is especially important for mobile users who may lose a signal while an app is in use.
- It has drag and drop support for elements.

Collectively, the new features of HTML5 support both Web application development and mobile Web 2.0. However, not all browsers have implemented it and it is problematic for older mobile devices. There are various sites that allow you to test your browser compatibility with HTML5 such as at <http://html5test.com/>.

As with all HTML, Web developers working in HTML5 have accessibility issues to consider. Over the years, WAI-ARIA (Accessible Rich Internet Applications) has attempted to fix accessibility issues in HTML. For example, HTML 4.01 uses scripting to create widgets to do things such as creating sliders in pages; ARIA provides element attributes that can inform various assistive technologies about the various roles of these elements. ARIA specifies roles for elements that inform assistive technologies about various document landmarks and document structure. ARIA attributes include examples such as application, banner, article, form, and navigation. Some of these ARIA attributes match HTML5 elements but some do not. Separate from this issue, many assistive technologies are not yet using HTML5, so the bottom line is that developers must be aware of these differences with the WAI-ARIA specification and HTML5.

SUMMARY

The huge success of the Web is largely due to the wide acceptance of HTML as a document format that presents graphically attractive content with useful hypertext links. This introduction to HTML described its origins, DTD, elements, tags, and structure. Examples of simple HTML files demonstrated the nesting rules and tags for linking, image references, framesets, and iframes. Some of these techniques, such as framesets, are now obsolete and better ways to control layout with CSS are discussed in the next chapter. Nonetheless, an understanding of link and base targets common in those designs is still useful. The need to create both absolute and relative references was discussed along with examples utilizing each approach.

The Web is going mobile, and specific adaptions of HTML have been developed such as WML and XHTML-MP. The newer HTML5 standard is gaining acceptance not just for mobilizing a site but also because it works better with multimedia, JavaScript, and CSS3.

REFERENCES

- Frain, Ben. (2012). *Responsive Web design with HTML5 and CSS*. Birmingham, UK: Packt Publishing.
- Gardner, L.D., & Grigsby, J. (2012). *Head first mobile web* (1st ed.). Sebastopol, CA: O'Reilly.
- Graham, I.S. (1996). *The HTML sourcebook* (2nd ed.). New York: John Wiley & Sons.
- Lawson, Bruce, & Sharp, Remy. (2011). *Introducing HTML5*. Berkeley, CA: New Riders.
- LeMay, L. (1997). *Teach yourself Web publishing with HTML 4* (2nd ed.). Indianapolis, IN: SAMS.
- Powell, T.A. (2003). *HTML and XHTML: The complete reference*. Emeryville, CA: McGraw-Hill/Osborne.
- Sivonen, Henri. (2013, March 9). Activating browser modes with doctype. Retrieved June 24, 2013, from <http://hsivonen.iki.fi/doctype/>
- Suetos, Shannon. (2010, April 26). HTML5: Worth the hype? Retrieved June 24, 2013, from <http://www.instantshift.com/2010/04/26/html5-worth-the-hype/>

202 Internet Technologies and Information Services

W3C Working Draft. (2011, April 5). HTML5 differences from HTML4. Retrieved June 24, 2013, from <http://www.w3.org/TR/2011/WD-html5-diff-20110405/#doctype>.

ADDITIONAL READING

Lehnert, W. (2002). *The Web wizard's guide to HTML*. Boston, MA: Addison-Wesley.

WEBSITES OF INTEREST

HTML 4.01 at <http://www.w3.org/TR/html401>.

XHTML at <http://www.w3.org/MarkUp>.

W3Schools HTML tutorials at <http://www.w3schools.com/html/default.asp>.

HTMLKit Free Editor at <http://www.chami.com/html-kit>.

Controlling Presentation with Styles

HTML was well suited to the initial needs of the Web, providing an exchangeable, structured, text-based file format for documents that could be enhanced with hypertext links and attractive formatting. The first views of the Web were strictly text based as were all available browsers. Even in this text-only environment, early Web browsers such as the NeXT, Viola, and Harmony clients provided some accommodation of style languages. In 1993, the graphical Mosaic browser arrived and made the graphically enhanced Web easily accessible to the public. It also resulted in HTML pages that were more attractive than those displayed with simple text-based clients; Web pages could have color, text could be styled, and images could be included and displayed. However, even this new graphical browser environment was stylistically limited; users could do little other than change display colors and fonts, and designers had limited control over the layout and appearance of their Web publications.

Controlling presentation has always been problematic in HTML. Presentation was typically dictated by the browser display defaults and the HTML publisher had to invest much time and effort in applying the procedural markup that is available in HTML to modify the display. Procedural markup could be used to specify a font family, its size, and its weight, but it had to be repeated many times throughout a document with different HTML elements. The presentation problem also gave rise to a variety of “hacks” that were used to accomplish desired effects. Tables were frequently abused as a way to control placement of HTML elements, and other odd tricks were often employed to overcome HTML deficiencies. For instance, there is no “tab” HTML tag, so the indentation of the first line of a paragraph might be accomplished by the use of a small transparent gif to act as an image spacer. Such devices worked, but they were not very elegant solutions to the layout problem.

As the Web grew and attracted more serious design efforts, the limitations of HTML for controlling presentation within standard browsers became more

apparent. One response by both Netscape and Microsoft to the increasing demand for better presentation control was to create new, proprietary HTML markup. Of course, the immediate problem with proprietary approaches is that those tags would not work across all browsers, making display in different environments problematic. Another approach was the development of style languages that could address presentation issues in a standardized fashion.

Just as in word processing, the creation of styles and style sheets for HTML has many obvious advantages. Style sheets allow the separation of content and presentation and represent a much more efficient way to control layout and appearance. This separation results in easier-to-manage websites because multiple HTML files can be controlled by a single set of style instructions in a style file. This centralized control of styles enables site-wide presentational changes by editing style information in just one location. Procedural approaches to formatting that require style information be inserted with each structural part of a document are time consuming and hinder global changes throughout a document or site. Another advantage of styles is the ability to deliver a view of the document tailored to the viewing device or user agent (i.e., client). Style directives provide preferred alternate views of the page in place of a default presentation. So in addition to being more efficient, styles allow for multiple views of the same content. Styles languages accomplish this without modifying the HTML standard by adding new, proprietary tags.

Style languages were an early Web development, and the development community was considering a number of competing proposals during that time. The approach that has been widely adopted is CSS. Style sheets were designed to “cascade” in order to accommodate the style preferences of the author and the viewer, as well as the capabilities of the display device and the browser. The origins of CSS are in the 1994 “Cascading HTML style sheets” proposal by Hakon Lie to the W3C; the reference to HTML was later dropped because CSS use is not limited to HTML (Lie & Bos, 1999). Since that time, Lie and others have developed CSS into a well-supported standard. There have been several iterations of CSS, and the current version is still Level 2.1, with Level 3 in development. Although CSS3 is not a complete, ratified version, it offers modules that are available for use now that are needed for responsive designs. The full specifications can be viewed at the W3C site (W3C, 2014). CSS3 modules are described later in this chapter, as are a few examples of their use.

The success of CSS and its wide acceptance is due to its relative ease of use and its wide support in all major browsers. Some style approaches require a full-fledged programming language, but the straightforward, declarative nature of CSS is quite accessible and easy to learn, and many excellent tutorials are available (McClelland, Eismann, & Stone, 2000; Powell, 2003; Frain, 2012; W3Schools, 2014). As with HTML and scripting languages, learning CSS really requires a hands-on approach. When teaching about these technologies, it is always a dilemma whether to provide background first and then go to examples or to start with examples first and then discuss broader principles. I have elected the former approach here, providing an overview before analyzing examples. However, some readers may prefer to look first at the examples provided and then come back to this overview. Either approach has its merits, but if you prefer the later, you may jump ahead to the examples section and return to the overview later.

During the “browser wars” of the 1990s, both Microsoft and Netscape promoted proprietary HTML, but when IE began to support CSS with IE version 3.0, Netscape came on board with its release of Netscape version 4.0. Although all major browsers support CSS, there are subtle differences in how each interprets and displays styled pages; these differences sometimes require special accommodation within the CSS code. Code modifications to accomplish browser specific effects are the “hacks” used to ensure consistent appearance across various browsers.

THE BOX MODEL

The Box Model is essential to understanding CSS layout control. This view of the visual space of the screen allows for the creation of a number of containers or boxes that can be arranged as desired within the page. As discussed in Chapter 7, many designs make extensive use of such an approach (see Figure 7.3). Each HTML container is given a unique identifier that is referenced in CSS rules, allowing certain styles to apply just those HTML containers. Each container created has margins, borders, and padding that can be adjusted to control the positioning of the box, the location of its content, and its appearance in the page. Each box has a border that can be visible or not. Padding is the space from the border to the start of the margins around the content. Margins are already familiar to almost everyone from word processing experience; they are the empty space around an area of content. Specific measurements can be set for left, right, top, and bottom spacing. In this model, boxes can be nested

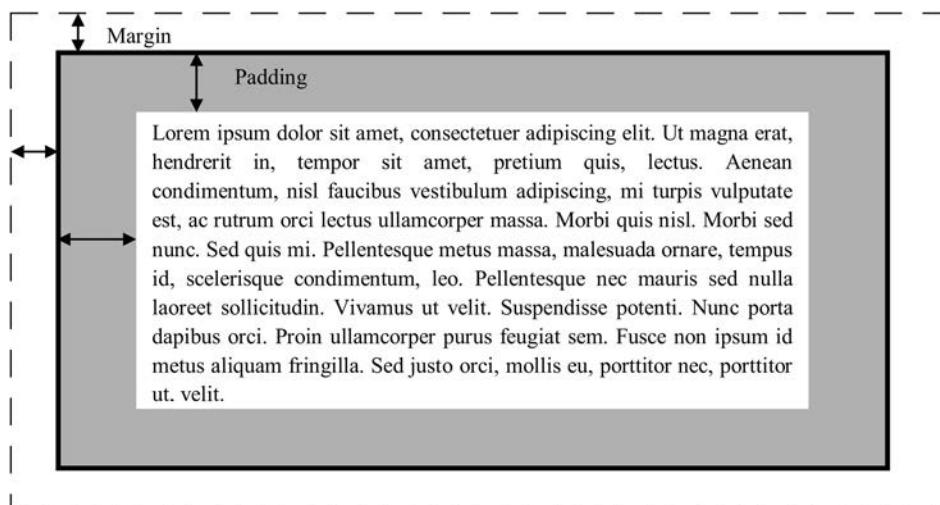


Figure 9.1 A box container with a black border. The margins surround the border and the gray area is the padding surrounding the content block, shown in white. The “Lorem ipsum . . .” paragraph is “pseudo-Latin” (some real words but used in a “jabberwocky” way); this is a common convention when “dummy text” is needed for layout purposes.

within boxes to achieve desired layouts. Figure 9.1 shows these various box attributes.

In HTML, block-level elements are always rendered on a new line, and inline elements flow across the page. Tables, lists, and forms are examples of block-level elements; images, links, and the actual content are examples of inline elements. Block-level and inline boxes are permitted in CSS. As mentioned in Chapter 8, HTML5 uses phrasing content and flow content for handling inline and block-level elements. The generic block-level HTML container is the **DIV** tag; the generic inline container is the **SPAN** tag. HTML5 introduced semantic tags for structural areas of a document such as headers, footers, sidebars, and navigation that can be used as CSS selectors instead of relying on generic DIV containers for them. A block-level box serves as the container for any boxes nested within it. The top-level container is the document window.

CSS POSITIONING

Boxes are positioned on the page with relative, absolute, or float directives, each with their own rules. In the absence of positioning directives, the normal flow is for block-level boxes to appear one after another vertically and for inline boxes to flow across the page from left to right, wrapping to new lines as needed. The two vertical margins are collapsed into a single margin that reflects the larger of the two values when boxes appear under other boxes.

Relative positioning starts with a box positioned with normal flow; the next box is then positioned using the offset values. Relative positioning can result in boxes overlapping other boxes and hiding content in another layer; how boxes overlap varies with different browsers. Absolute positioning forces the container to appear in a specific location on the page. Fixed positioning is a form of absolute positioning that forces the box to appear in the same place relative to the browser window; fixed locations do not change as the user scrolls through a page. This is useful when you would like to ensure that a box, such as one with a navigation menu, is always visible in the same place on the screen. The FLOAT directive controls the location of the box and aligns it either at the top left or top right of another box; the floated box appears as would any other content within the container it is placed. The inline content of the parent box then flows around the floated box. Examples of these types of positioning are provided later in the chapter.

ADDING STYLES TO HTML

There are many ways to incorporate style information into HTML. When no style information is given within the HTML, the Web page appearance is determined by the browser defaults and the user's preference settings within the browser. For Web designs using CSS, many options are available to add style information. Style information can be placed within a separate file and then referenced within the HTML document, or it can be embedded within the HTML container itself. A common way to place style information within the HTML is with the addition of the tag pair in the head area that contains the set of rules

to be referenced. Within the style tag, it is possible to import style information from some external source to add rules to those written in the local file. It is also possible to add inline styles through the addition of a style attribute within an HTML element itself. How each of these approaches are different, and how rule conflicts are resolved if more than one rule is available for an element, is discussed next.

External Style Files

Putting style information into a separate file from the HTML file itself has several practical and design advantages. By completely separating style information from the content, the HTML code is kept free of added coding that clutters the content and HTML code of the page. In addition, if a style change is desired, updates made in that single file will then apply to every HTML file that references it; the alternative requires style changes be made in every individual HTML file that needs the change.

To use an external file, a text file must be created containing the rules that would have been placed within the style tag pair in the HTML file. This style file has a “**CSS**” extension and it should contain nothing other than the rule set. There must not be any extraneous text or markup, including the beginning or ending tag, in this file. To reference such an external style file, the HTML **<link>** tag is placed in the head area as shown in the following example:

```
<link rel = "stylesheet" type = "text/css" href = "mystyle.css" />
```

The relation and type attributes identify this as a preferred style sheet created in a text format. Note that the **REL** attribute can also refer to an alternate style sheet as well. In addition, multiple **LINK** tags can be used in the head area; for instance, a separate style file might be used for the less commonly used HTML elements. The **HREF** attribute value points to the location of the file with the style rules.

The **LINK** tag also allows for an optional **MEDIA** attribute that has a defined set of possible values, some of which are shown in Table 9.1. (Note that not all browser versions handle media attributes.)

TABLE 9.1 Link Media Attributes

Media Attribute Values	Purpose
Screen	Default media setting; for computer displays
Print	Optimize for a printer
Projection	For projection displays
Aural	For screen-reading programs

Embedded Styles

To embed styles, the tag is placed in the head area and serves as a container for the rule set. Some or all the rules may be imported from the other source file with the “**@import**” statement. There are two equivalent ways to structure the **@import** statement:

@import “somestyle.css” OR @import url(“somestyle.css”)

These **@import** statements can coexist with the specific rules that are in the style container within the head area. Given that there are multiple sources for CSS rules that could potentially give conflicting instructions for the presentation of the same elements, there are rules about what rules “trump” other rules; the specifics of those rules are discussed later in this chapter.

Inline Styles

Inline styles refer to style attributes added to an element itself. Control at this level potentially requires the micromanagement of every element in the document, and it is usually only used to override a broadly applied style that an element would otherwise inherit from a higher-level style. For example, the alignment of text for a specific paragraph could be set with a style attribute placed in the paragraph tag itself as in the example: **<p style = “align: right”>**.

CSS SYNTAX AND RULES

Languages have vocabulary and syntax, and CSS is no exception. The vocabulary is the defined set of selectors and properties; the syntax dictates how each CSS statement is structured and the punctuation needed for successful parsing by the displaying program. CSS applies style information to HTML elements by the formulation of rules. Style languages also permit the inclusion of comments. Comments in CSS begin with a forward slash and an asterisk and end with an asterisk and a forward slash. Comments may be used to explain what the style sheet is intended to do or to provide statements of authorship or terms for shared use. In addition, it is common to use comments as a way to deactivate code without actually removing it; this is referred to as “commenting out” a code block. This use of comments is a helpful technique for troubleshooting style rules. It is also an effective learning technique that allows you to turn rules off or on by adding or removing comment designations and then observing the effects on document presentation.

CSS Rules

CSS rules have a selector and a declaration. The selector identifies one or more HTML elements where the style will be applied. Declarations specify properties and values. The declaration can consist of one or multiple property/value

```
/* this is a comment; a rule is on the next line*/
H1 {color: red}
```

Figure 9.2 A style tag with a simple CSS rule. Here H1 is the *selector*, and the portion in curly brackets is the *declaration*. The declaration consists of a *property* (in this example “color”) and a *value* (in this example, “red”).

```
H1, H2, H3 {
  color: red;
  font-weight: bold;
  font-family: helvetica;
  font-variant: normal;
  font-style: normal;
}
```

Figure 9.3 A rule with multiple selectors and declarations.

pairs. The set of properties consists of the 50 or so style attributes that determine the presentation of an HTML document; the property is specified by assigning a value to it. Declarations are surrounded by a set of curly brackets. A property is separated from its value by a colon, and multiple property/value pairs within a single declaration are separated by semicolons. CSS rules are embedded either within a style tag container in an HTML document or in a separate CSS file; an example of a simple CSS comment and rule is shown in Figure 9.2.

In this rule, **H1** is the selector, and the declaration, within the curly brackets, has a property (color) and an assigned value (red). This rule selects all the heading level 1 elements in the document and instructs the browser to present them in red text.

In Figure 9.3, there are three selectors: **H1**, **H2**, and **H3**. Multiple selectors are grouped together and separated by commas. The declaration has five property/value pairs; each pairing is separated by a colon, and each declaration is terminated with a semicolon. Wildcards can also be used as a way to apply a declaration to all possible selectors.

Rules can reference elements that are nested within other elements. Figure 9.4a shows a rule with a hierarchical selector string; for example, the selector “**ul li a**” selects only anchor tags that are within list items within any unordered list. Figures 9.4b and 9.4c show rules with an “**id**” selector, which is indicated by a pound sign (#) either at the beginning of the selector string (9.4b) or after an initial selector (9.4c). The pound sign convention is used as a way to bind a style to a particular **<div>** container in the HTML file that has been given a matching “**id**” attribute. (Note that “**id**” name values should not begin with a number, as this may not work in all browsers.)

CSS Classes

A group of declarations can be associated with a **style class**. A class can be independent of any selector, or it may be associated with a specific element

210 Internet Technologies and Information Services

Example	Rule	Action
a.	ul li a {color: red}	This would select anchor tags (links) within list items that are part of any unordered list.
b.	#nav ul li a {color: red}	Selects a DIV container with the id of “nav” and then applies the rule to any anchor tag that occurs as a list item within unordered lists, but only to those within that DIV container.
c.	p#first {text-indent: 2 em}	Selector is paragraph, but application of the rule is restricted only to paragraphs that have the attribute id = “first.”

Figure 9.4 Three example rules and their actions.

```
p.new {  
color: red;  
font-family: arial;  
}
```

Figure 9.5 A class associated with a selector; selector is “p” and class name is “new.”

In the previous example, the class “new” is associated with the paragraph element with the “p” selector. To create an independent class not associated with any particular element, the selector would be dropped as shown in Figure 9.6.

Just creating a class by itself does not change any aspect of the presentation; it simply is available for possible use. To see the class in action, it must be referenced as a class attribute added to some element. If a class is created with a selector, it is applied only to that element, whereas an independent class can be associated with any element. For instance, to apply the two classes shown in Figures 9.5 and 9.6, each could be referenced in some HTML code as shown here:

```
<p class = “new”>  
<H1 class = “new”>
```

Note that either class definition could be applied to the paragraph element. However, only the independent class definition would work as applied to the **H1** shown previously; classes with selectors are applicable to only that designated

```
.new {  
color: red;  
font-family: arial;  
}
```

Figure 9.6 A class definition without a selector.

element. Independent classes are more powerful and flexible, but classes with selectors clearly identify the intended element to be associated with the class. Both selector and nonselector classes are used in CSS and the decision between defining selector-based or independent classes is determined by designer preference and the organization of the style sheet.

Pseudo-Classes

```
a:link {color: red}
a:visited {color: blue}
a:hover {color: green}
a:active {color: black}
```

Figure 9.7 Pseudoclass use with anchor properties.

```
a.new:visited {color: blue}
```

Figure 9.8 A class used with a pseudoclass.

A pseudo-class is similar to creating a class except it applies to different aspects of the same element that are to be distinguished from each other, as opposed to different elements themselves. For example, the anchor tag is used to create hyperlinks, but you may wish to stylistically distinguish unvisited links, visited links, selected links, or hovered-over links. Pseudo-classes enable stylistic control of the various forms of a single element; these rules use a colon between the element and the specific element form. Pseudo-classes that are frequently defined for the anchor tag are shown in Figure 9.7.

Pseudo-classes can be combined with classes; for instance, the independent class created earlier called “new” could be specified in a pseudo-class, as shown in Figure 9.8.

Sizing and Color Units in CSS

It is often necessary to specify sizes and positions in CSS, and these measurements can be provided in absolute, relative, or percentage terms. Absolute units include English and metric measures, or inches, centimeters, and millimeters, as well as typographic measures of points and picas, which are respectively, 1/72 of an inch and 12 points. Absolute measures are more relevant to the world of print and are problematic for most Web designs because the designer cannot anticipate the screen sizes of users. Even pixels (a contraction of “picture elements” and abbreviated as “px”) are not absolute units because their size is dependent on screen size, dot pitch, and the resolution setting of the hardware in use.

```
body {
margin-left: 20%;
font-family: Verdana;
font-size: 12 pt;
line-height: 125%;
}
```

Figure 9.9 Using percentages as units of measure.

Percentage units are frequently used to control the amount of space allowed for elements within the browser window and are particularly important for responsive, fluid layouts. Percentages can refer to a portion of the full screen or relate to a percentage of some other set measurement, most often the font size. The rule in Figure 9.9 has the body left margin set to 20 percent of the screen real estate and the line height set to a percentage of the chosen font size.

Regardless of the browser window size, the left margin as specified in this example would stay proportionately the same, and the line height, which controls line spacing, would be 125 percent of the font size. Using a percentage in the rule allows the spacing to remain proportional even if a user setting forces a different font size in the browser preferences. Note that the font declarations can be combined into one statement specifying the font size and line height by giving two point sizes separated by a slash, as in “**font: 12/15 Verdana.**”

Relative measures include **px** for pixel and the **em** designation for a single character height. As noted earlier, pixels are relative measures because their sizes are dependent on a number of factors outside the designer’s control. For instance, specifying a left margin of **40 px** looks quite different when viewed with screen resolutions 640 x 480 vs. 1280 x 024 or when viewed on a 15" vs. a 20" monitor. The **em** unit is relative to the font size and equal to the height of a character. Again, the relative nature of this unit helps keep spacing proportional to the font if the font size in the style is overridden by the browser.

Although pixel sizes are relative, pixel-based elements and pixel-based typography should be converted to their respective percentage or **em** equivalent to achieve a fluid layout in a responsive design. For example, a final design layout in Photoshop (referred to as a “comp”) might call for special base and header fonts expressed in pixels, but these need to be converted to more flexible **em** measures for a responsive design. The design comp might define the base font at 16 px and have a level one header set to 24 px, both common browser defaults. However, these will look different in various viewports and other rules applied at the body level might be inherited that could constrain the header to the base font size. The solution Marcotte (2011, p. 20) suggests is to convert to **em** measures based on a simple formula:

$$\text{Target} \div \text{context} = \text{result}$$

So in this example, if the desired target size for the header is 24 px and the context is the 16 px base font, to convert to **em** the target divided by the context to give 24/16 or 1.5, so the rule in **em** would become:

H1 {font-size 1.5 em}

This sets the header size appropriately proportional at one and a half times the base font size. Marcotte also suggests this approach to creating more flexible rules is applicable to converting dimensions given in pixels to percentage values for box sizing when building flexible grids; an example of this technique is examined later in this chapter.

Color is also a property that can be specified in style rules using the RGB color model discussed in Chapter 7. The units that specify a color in this model are numeric representations of the amount of red, green, and blue, with 8 bits allowed for each additive primary color. Color values can be specified by name (e.g., red, blue, yellow), as percentages, or numerically with either decimal (0–255) or hexadecimal (0–ff) notation. The color white could be specified in any of the following equivalent ways:

```
color: white  
color: rgb(255, 255, 255)  
color: rgb(100%, 100%, 100%)  
color: ffffff
```

Hexadecimal is a common way to specify color in both HTML and CSS. If each pair of HEX digits is the same, the notation can be shortened to specifying only the first digit of each pair. So “**ffffff**” can be simplified to “**fff**.” If only three digits are given, the omitted digit from each pair is assumed to be the same as the first when the rule is parsed.

UNDERSTANDING THE CASCADE

There are multiple entities involved in the control of the presentational aspects of a Web page. Obviously, the author/publisher cares about the appearance of the site, but the viewer wants, and has, some level of control. In addition, the device or software in use plays a role in page appearance and style application. The *cascade* part of the name for CSS reflects that these various overlapping directives interact and may result in multiple directives for the same HTML element. A defined order of precedence is necessary to resolve rule conflicts; generally, rules that are local to the HTML are preferentially enforced over ones that are from sources that are more distant. At the lowest level, if there are no style rules, the browser controls presentation. However, if an external style sheet is referenced with a **LINK** tag those style directives override the browser defaults. Next, if a STYLE tag is in the head area with rules that apply to an element, those local rules override any external style sheet directions for that element. If a local STYLE container in the HTML uses both imported and local rules for the same element, the local rules “trump” imported ones. Inline styles using a style attribute in an element override all other conflicting style information.

The styles applied to a specific element depend on the source of the style information for it. If an element is not specifically identified as a selector in a rule, the element's style properties are inherited from the next higher HTML element. For example, if there are no selectors for paragraphs, paragraphs then simply inherit the appearance properties specified for the BODY element. If there are rule declarations for some specific element and property, they are sorted by origin and weight. If there are two or more declarations for the same

element, any explicit weighting is then factored in. For instance, a declaration can be given higher weighting by the inclusion of the **!important** designation, at least with browsers that respect it. The declaration in the following rule will be given higher weight in the precedence calculation by most browsers because of the **!important** designation.

```
#nav {margin-right: 1em !important}
```

After factoring in explicit weighting, the browser then sorts the rules by origin as described previously. To recap: Rules presented in style attributes for the elements themselves override rules in a more distant **<style>** tag; local rules within a **<style>** container override those imported into it with the **@import** notation; all these rules override those imported from an external style sheet file with the **LINK** tag. If two rules have the same level of weighing and origin, specificity is given precedence over rules that are more general. Finally, if this secondary sort does not resolve the rule conflict, the last occurrence of the rule is enforced over an earlier rule.

HTML5 AND CSS3 MODULES

HTML5, discussed in the previous chapter, can be combined with the CSS3 modules that are presently available to produce responsive designs. The basics of each are unchanged but a few of the specific features and differences are relevant to the way it works with CSS. In HTML5, the generic **<div>** container is still used, but only when there is not a specific HTML5 element available for the needed container. Instead of the older approach of creating a generic container such as **<div id= “nav”>**, the HTML5 semantic tag **<nav>** is used directly. In addition, such HTML5 container elements can be used multiple times in a single page; for example, different sections of the page can each have header and footer elements nested within them.

CSS rules reference HTML **div** containers by a matching **id** reference in an HTML file, such as the one shown in Figure 9.16. As shown in the CSS in Figure 9.4, the ID references can be selectors and appear with a beginning pound sign followed by the id value, for example **#nav**. With HTML5, CSS reference the element name directly as with any selector as in this example:

```
nav {float: left; width: 20%;}
```

The hierarchy of selectors remains the same with CSS3; for example, a style rule for an HTML5 header element in an article element would be:

```
article header {property: value}
```

CSS3 Modules

CSS has been divided into various components called modules; these include much of CSS 2.1 along with new modules for CSS3. Even though it is

TABLE 9.2 Some New Features in CSS3

@font-face	Allows fonts to be downloaded from a server to better control how they are rendered
Opacity values	Controls the opacity of an element from transparent to opaque
RGBA	Adds transparency options to color specifications
Border-radius	Allows for rounded corners to boxes
Box-shadow	Allows for drop shadows on elements
Text-shadow	Allows for drop shadows for text
Transform	Allows objects to rotate or skew
Multiple background images	Allows multiple image layers in backgrounds

not a fully ratified version of CSS, these new modules are available for use now. The good news is that incorporating CSS3 features into the design will not “break” it—browsers that do not recognize it will simply skip over these directives, allowing designers to embrace the new modules for browsers that support it (Frain, 2012). CSS3 includes animation and transition effects that allow elements to move to new locations or rotate in various ways, replacing JavaScripts or the waning Flash technology used for these effects. Especially when combined with HTML5, it can be used to support responsive designs that work well across devices. Jacobs (2011) provides a “Top 10” list of new features and Table 9.2 lists some of these new CSS3 features.

CSS HACKS AND SHORTCUTS

As mentioned earlier, there are a number of hacks used in CSS to compensate for the variations among different browser interpretations of style rules. A hack refers to either an added rule or an odd application of a rule in order to achieve consistent page appearance across different browsers. The W3C Box Model defines the height and width of a box as that of the content area

Rules that could result in two different box sizes depending on the browser	A sample “hack” to address this difference
#nav { width:100px; padding:10px; border 5px; } 	#nav { border: 5px; padding: 10px; width: 100px !important; width 130px; }

Figure 9.10 Box size definitions that require hacks with some browsers on the left; a hack fix on the right.

Multiple property/value pairs in a declaration to individually specify margins	Single property with four values in the declaration to specify the four margin settings
{ margin-top: 1em; margin-right: 2em; margin-bottom: 3em; margin-left: 4em; }	{ margin: 1em 2em 3em 4em; }

Figure 9.11 Size designations on left and shortcut method to specify all in one statement on the right.

only. However, IE prior to version 6 adds in the padding and border space to the overall dimensions of the box, resulting in different rendering than Firefox or later IE versions. For instance, consider the rules shown in the left side of Figure 9.10.

In most browsers, the rules on the left result in a “box” that is 130 pixels ($100 + 10 + 10 + 5 + 5$). However, in some versions of IE the full box size would be constrained to the content width given as 100 pixels, resulting in a smaller box. One fix is to address this by creating a “box within a box” and set the padding and borders to zero for the outer box. Another fix, shown on the right side of Figure 9.10, is a hack made possible by a difference between IE and other browsers regarding rule weighting; the **!important** designation is ignored by some versions of IE. Normally the **!important** weighting results in giving the rule precedence over a later one, except in browsers that do not recognize this designation. Browsers that comply with the W3C definition of box height and width and that pay attention to the **!important** weighting would make the box $100 + 10 + 10 + 5 + 5 = 130$ pixels. The versions of IE that would make the full box size equal to the width dimension ignore the **!important** weighting. In that case, IE would default to the precedence of the “last rule,” which gives the width as 130 pixels, the same box size as the other browser. Although such hacks can be useful, the best practice is to avoid them as much as possible.

There are also some useful shortcuts in CSS. The color value shortcut, already mentioned, allows a single HEX digit to stand in for the double digit if the values are the same. Specifying a color as “**3cc55**” is equivalent to just using “**3c5**” to determine the amounts of red, green, and blue, respectively. In addition, when specifying margins and padding, a single rule can list all four values with the assumption that the order is always top, right, bottom, left. Therefore, the declaration shown on each side of Figure 9.11 are equivalent.

RESPONSIVE DESIGN AND CSS

Important developments for responsive designs include enhanced media queries, flexible grids, and fluid images. Web pages are viewed in different environments, referred to as the *viewport*, and these have different sizes and

resolutions. The desktop viewport is the browser window, which may be full screen or resized within that space. Smartphones have much less screen real estate; common sizes were discussed in Chapter 7.

In the preceding discussion of layout, boxes are displayed in different browsers and all the measurements are in pixels. However, as previously discussed, pixels are problematic for a number of reasons, especially on small, mobile viewports. This discussion is further complicated by the fact that there are two ways to think about pixels when a Web page is displayed. The hardware determines the size and resolution of the device, but CSS defines pixels that may differ from that; for example, when a user zooms in or out, these relative pixel sizes change even though the hardware determined pixels have not. Sizing of all elements must be made flexible and responsive to ensure appropriate display across different viewports.

Media queries represent one way to control the appearance of a site in different viewports. Media queries allow the page being displayed to adapt to the properties of the viewing environment such as its resolution or type. A media query consists of a media type and expressions in the form of true/false questions; when a media type matches and the conditions in the expression are true, the CSS rules associated with it are applied. Media types were mentioned earlier as an attribute of the LINK tag, but CSS also has the “**@media**” rule that can associate different style rules with different media within the same stylesheet. For example, examine the rule below:

```
@media screen and (min-width 320px) . . . {one or more CSS
declarations follow here }
```

This rule has two conditions connected with a Boolean AND; it first checks that the display is a screen, and if so, that it is at least 320 pixels wide. If both these conditions were true, then the style declarations that follow would be applied.

Images are another issue in mobile design because if not constrained to fit within their intended container, they can overflow the intended container boundaries and obscure other boxes in the design. Images can be made fluid to prevent them from displaying in their native resolutions, which are often larger than the box in which they are expected to display. A simple way to constrain an image to its appropriate view size is with the CSS rule:

```
img {max-width: 100%}
```

This forces the image to fit within the size of the container and not exceed it; this rule could be expanded to include other media selectors such as embedded videos as well. One hack related to this is that IE6 and older do not support “max-width” and instead need the property “width.” However, these are not interchangeable because it changes the meaning of the rule slightly; the first prevents the maximum image width from overflowing the container, the latter forces the image to be exactly 100 percent of the container. Another issue with fluid images is that some Windows browsers do not always scale the CSS resized images well (Marcotte, 2011).

Responsive design strategies also can include flexible grids based on the multiple column feature of CSS3. Print designers use grids to facilitate the layout of the content to the size of the page. For Web design, these are made flexible to support both user browser preferences and the viewport itself. There are CSS grid system templates available online, for example, at www.columnal.com, but simple grids can be assembled with CSS and proportional sizing. The same rule of target/context = result can convert pixel-sized container widths in a design comp to flexibly sized ones using percentages. For example, say a comp drawing defines the containers shown in Figure 9.12. The

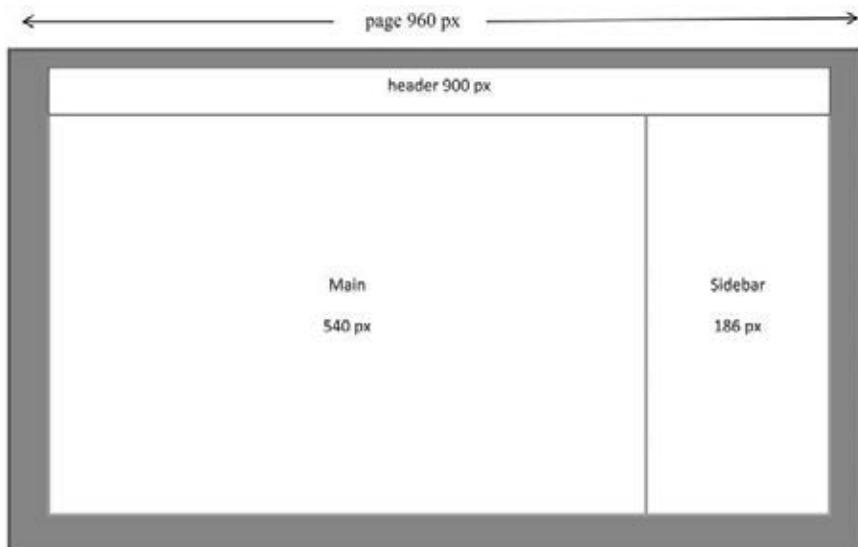


Figure 9.12 A comp drawing that defines content containers in pixels (margins and padding for each container not shown).

Fixed pixel CSS	Flexible responsive CSS equivalent
#page { margin: 30px auto; width: 960; }	#page { margin: 30px auto; width: 90%; }
#header { margin: 0 auto 50px; width: 900px; }	#header { margin: 0 auto 50px; width: 93.75%px; /*900px / 960px */ }
#main { float: left; width: 540px; }	#main { float: left; width: 60%; /*540px / 900px */ }
#sidebar float: right; width: 186px; }	#sidebar float: right; width: 20.66666%; /* 186px / 900px */ }

Figure 9.13 Fixed vs. flexible layout rules.

overall canvas size is 960 pixels, the section main is 540 pixels, and sidebar is 186 pixels. These measures could be left fixed in pixels in the CSS, but that would not be flexible, resulting in vertical and horizontal scrollbars on smaller viewports.

Marcotte (2011) suggests converting these pixel dimensions to proportional percentage values using the same target/context formula described earlier. Figure 9.13 shows both approaches in the CSS for the layout in Figure 9.12. The percentage rules show the values used in the calculation as CSS comments.

PUTTING IT TOGETHER: A CSS EXAMPLE

The best way to learn many of these techniques is by looking at examples and applying them with hands-on practice. The following example of a CSS style file and its action on a simple HTML document is provided to illustrate how CSS can control presentation. However, looking at examples alone is no substitute for practice you do on your own. The following HTML file, rendered in the Firefox browser with styles turned off (there is a useful developer toolbar for Firefox that makes it possible to easily turn styles and scripts on or off), shows a nonstyled HTML with lists of links. The table and final footer areas have no special formatting in this view, shown in Figure 9.14. The same file

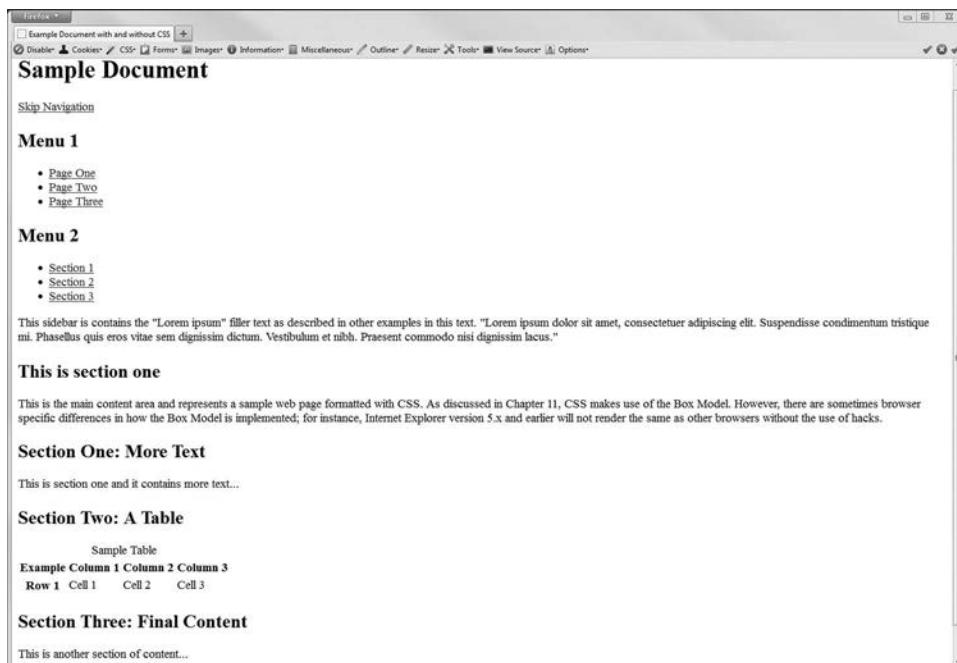


Figure 9.14 A screen shot of the HTML file from Firefox with styles turned off preventing the application of any style directives.

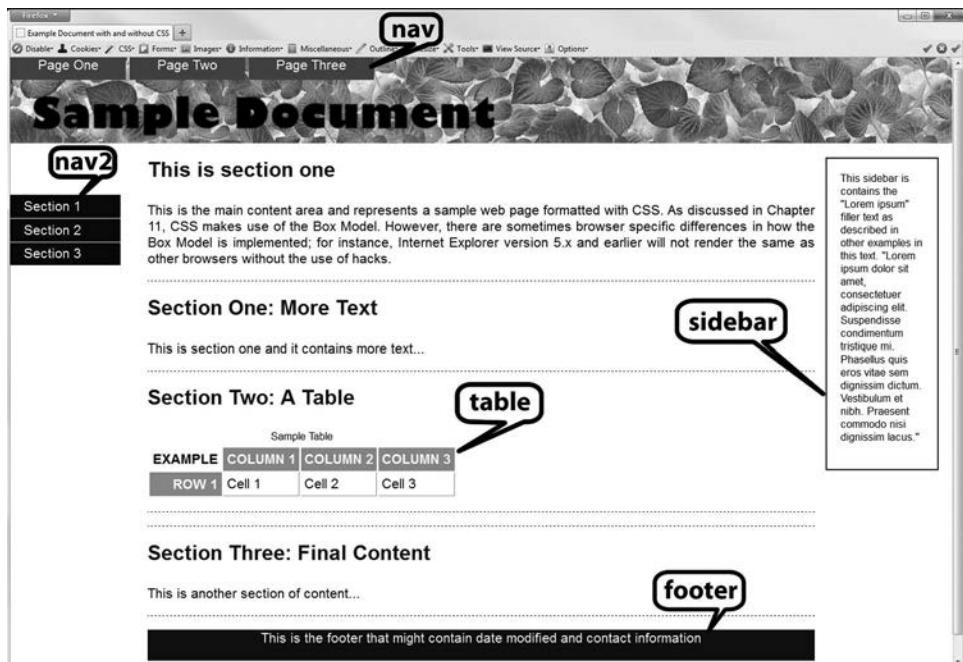


Figure 9.15 The same Web page as in Figure 9.14 with styles enabled.

rendered in the FireFox browser with styles turned on and utilizing a CSS file is shown in Figure 9.15.

Note that the two menu lists have been positioned and formatted to be readily recognized as navigation areas; in addition, the background color of each menu item will change with a hover over to make them more interactive and animated. Other layout controls have made some text positioned as a sidebar, provided table formatting, and made the footer area appear centered and in color, making an appropriate area for contact information and a last modified date. The HTML code that is the source of both views of this page is shown in Figure 9.16. The callout notes inserted in the screen shot in Figure 9.15 give the names of the DIV containers that are referenced in both the HTML and the CSS files. These are explained in the series of figures that follow Figure 9.16.

The following figures present specific rules in the CSS file that change the HTML file's appearance to that shown in Figure 9.15. The rules all reside in the separate file **style-example.css**, which is referenced in the HTML file LINK tag. For purposes of explanation, the rules from this file have been segregated into those that do not reference a specific HTML DIV container and those that do. The rules are listed on the left of each figure, and the explanation of their action is on the right. Figure 9.17 has a list of the CSS rules that do not have a **DIV** container specified.

The following figures are rules associated with **DIV** containers. Figure 9.18 has the rules for the "header" and "footer" **DIV** containers. Figure 9.19 shows

```

<html>
  <head>
    <title>Example Document with and without CSS</title>
    <link rel="stylesheet" type="text/css" href="style-example.css" />
  </head>
  <body>
    <div id="header">
      <h1>Sample Document</h1>
    </div>
    <div id="skipnav">
      <a href="#content">Skip Navigation</a>
    </div>
    <div id="nav">
      <h2>Menu 1</h2>
      <ul>
        <li><a href="pageone.htm">Page One</a></li>
        <li><a href="pagetwo.htm">Page Two</a></li>
        <li><a href="page3.htm">Page Three</a></li>
      </ul>
    </div>
    <div id="nav2">
      <h2>Menu 2</h2>
      <ul>
        <li><a href="#sec1">Section 1</a></li>
        <li><a href="#sec2">Section 2</a></li>
        <li><a href="#sec3">Section 3</a></li>
      </ul>
    </div>
    <div id="sidebar">
      <p>This sidebar contains the "Lorem ipsum" filler text ...</p>
    </div>
    <div id="maincontent"> <a name="content"></a>
      <div id="sec1">
        <h2>This is section one</h2>
        <p>This is the main content area ...</p>

        <h2>Section One: More Text</h2>
        <p>This is section one, and it contains more text...</p>
      </div>
      <div id="sec2">
        <h2>Section Two: A Table</h2>
        <div id="table">
          <table summary="Sample Table"> <caption>Sample Table</caption>
            <tr> <th class="title">Example</th>
              <th>Column 1</th>
              <th>Column 2</th>
              <th>Column 3</th> </tr>
            <tr> <th class="row">Row 1</th>
              <td>Cell 1</td>
              <td>Cell 2</td>
              <td>Cell 3</a></td> </tr>
          </table> </div>
        </div>
        <div id="sec3">
          <h2>Section Three: Final Content</h2>
          <p>This is another section of content...</p>
        </div>
      </div>
      <div id="footer">
        <p>This is the footer that might contain date modified and contact information</p>
      </div>
    </div>
  </body>
</html>

```

Figure 9.16 The HTML file code for the Web page displayed in both Figure 9.14 and Figure 9.15. Note the DIV container names.

the rules for the “nav” container, and Figure 9.20 has the “nav2” **DIV** container rules. Figure 9.21 shows the “maincontent” and “sidebar” div rules. Finally, Figure 9.22 has the rules for the “table” **DIV** container.

With these simple rules, the HTML file is transformed in appearance, but if the styles are not desired or are unavailable in a particular viewing environment, an alternate version of the page is available that presents the navigation links as linear lists followed by the content of the page.

CSS Rules not associated with a DIV	Notes
* { margin : 0; padding : 0; }	The * is a wildcard for all element selectors, and the declaration starts the new page with all container margins and padding set to zero.
h2, p, ul, ol, table, fieldset { margin-bottom : 1em; }	This will set the bottom margin for the selectors listed (fieldset is for forms).
body { font-family : arial, helvetica, sans serif; }	This lists the preferred font for all body content; fonts are listed in order of preference. For instance, if Arial is not available, Helvetica will be used; if neither are available, any sans serif font will be used.
a { text-decoration : none; }	This turns off the default link formatting for anchor tags.
#skipnav { display : none; }	This will suppress the display of the “skipnav” DIV container. However, if CSS is not enabled, the skipnav area will still be available.

Figure 9.17 Rules not associated with a DIV container.

DIV name	CSS Rule	Notes
header	#header h1 { background : url(banner.png) top left no-repeat; } #header h1 { display : block; height : 100px; text-indent : -9999px; }	All these rules apply to the header DIV container and will replace the H1 content with an image. A nonrepeating background image is set for this area, and within that area, the H1 content will be set off screen with a negative indent but remains available when viewed without CSS.
footer	#footer { clear : both; } #footer { color : #fff; background-color : #006; text-align : center; }	This clears a space for the footer area and prevents any floating containers from getting in the way. Text and background colors are set, and text is center aligned.

Figure 9.18 CSS rules for the header and footer DIV containers.

DIV Name	CSS Rules	Notes
nav	<pre>#nav { position : absolute; left : 0; top : 0; } #nav h2 { display : none; } #nav ul { list-style-type : none; } #nav ul li { display : inline; } #nav ul li a { padding : 0.5em 2em; text-decoration : none; color : #fff; background-color : #555; } #nav ul li a:hover { color : #006; }</pre>	These rules apply to the nav DIV container. The container is positioned, and specific selectors within the container are formatted as follows: H2 elements will not display; unordered list formatting is removed and list items made to flow across the screen with the inline display; and links within the list items are formatted by removing default formatting and specifying text, background, and hover colors.

Figure 9.19 The rules for the “nav” container.

DIV Name	CSS Rules	Notes
nav2	<pre>#nav2 { position: fixed; float : left; width : 7em; top: 10 em; } #nav2 h2 { display : none; } #nav2 ul { list-style-type : none; } #nav2 ul li a { display : block; width : 100%; padding : 0.25em 0 0.25em 1em; background-color : #004; border-bottom : 1px solid #eee; color : #fff; text-decoration : none; } #nav2 a:hover { background-color : #777; }</pre>	These rules apply to the nav2 DIV container. The container is floated to the left side, and the fixed positioning declaration keeps this navigation area in the same screen location of 10 em from the top regardless of page scrolling. The container width is specified as 7 em. Within this container, H2 elements are suppressed, unordered list formatting is removed, and links within list items will be reformatted in a block display with padding, colors, and borders specified.

Figure 9.20 The rules for the “nav2” container.

DIV Name	CSS Rules	Notes
maincontent	<pre>#maincontent { margin : 0 9em 0 9em; } #maincontent { padding : 1em; text-align : justify; } #maincontent div { border-bottom : dashed 1px #006; margin-bottom : 1em; } #maincontent h2 { color : #006; }</pre>	These rules apply to the maincontent DIV container and specify margins, padding, and text alignment. DIV containers within the maincontent area will have borders, and the color of H2 elements is set.
sidebar	<pre>#sidebar { float : right; width : 6em; margin-top : 1em; margin-right : 1em; } #sidebar { padding : 1em; border : solid 1px; } #sidebar p { font-size : smaller; text-align : left; }</pre>	These rules apply to the sidebar DIV container, which will float to the right and be set at 6em wide. The padding, borders, and paragraph text formatting are all specified.

Figure 9.21 The rules for the “maincontent” and “sidebar” containers.

DIV Name	CSS Rules	Notes
table	<pre>#table caption { padding-bottom : 0.25em; text-align : center; font-size : smaller; } #table th { padding : 0.25em; text-transform : uppercase; color : #fff; background-color : #999; } #table th.row { text-align : right; } #table th.title { text-align : center; background-color : #fff; color : #000; } #table td { border-right : solid 1px #999; border-bottom : solid 1px #999; padding : 0.25em; } #table tr:hover { background-color : #ccc; }</pre>	These rules will control the presentation of the table DIV container. The caption, headings, title, and data cells are all formatted. In addition, a hover color is added to highlight a row when the cursor passes over it. Note that this hover-over effect for table cells may not work in all browsers.

Figure 9.22 The rules for the table DIV.

SUMMARY

CSS is a widely accepted standard for presentation and allows designers to have much better control of their websites. The separation of presentation from content is an efficient strategy for managing style changes. CSS can also give the user some control of presentation by allowing for different views of the content that can accommodate the limitations of different devices and browsers. Given the growing importance of the mobile Web, learning CSS is especially important to creating responsive designs that can accommodate different viewports, and HTML5 combined with CSS level 3 provides new tools to accomplish this. CSS is easy to learn, but as with many technologies discussed in this text, hands-on experience is essential to become truly proficient in its application.

REFERENCES

- Frain, Ben. (2012). *Responsive Web design with HTML5 and CSS*. Birmingham, UK: Packt Publishing.
- Lie, H. W., & Bos, B. (1999). *Cascading style sheets, designing for the Web*. Reading, MA: Addison Wesley.
- Marcotte, Ethan. (2011). *Responsive Web design*. New York, New York: A Book Apart.
- McClelland, D., Eismann, K., & Stone, T. (2000). *Web design studio secrets* (2nd ed.). Foster City, CA: IDG Books.
- Powell, T. A. (2003). *HTML and XHTML: The complete reference*. Emeryville, CA: McGraw-Hill/Osborne.
- W3C. (2014). Cascading style sheets home page. Retrieved January 22, 2014, from <http://www.w3.org/Style/CSS>.
- W3Schools. (2014). CSS tutorial. Retrieved January 22, 2014, from <http://www.w3schools.com/css/default.asp>.

This page intentionally left blank

Introduction to Web Programming

Web publishing can be done using various CMSs, which will be discussed in Chapter 11, or directly in HTML with or without CSS styles, as discussed previously in Chapters 8 and 9. Static content residing in separate HTML files comprises a large portion of existing Web resources, and using HTML alone has limited options for interactivity. Further, static approaches require that content updates take place at the individual file level, which is problematic for sites with a large number of pages to manage. One alternative is to utilize programming, often in conjunction with databases, to create more dynamic and interactive websites. Dynamic content does not exist in some predefined HTML container but is generated on demand by a query or script action; the combination of database and script programming technologies are the basis of the many CMSs, such as Drupal. Such dynamic approaches have many benefits, but one consequence of this approach is that search engines are usually unable to index these dynamic pages because they do not exist as autonomous files residing on a Web server; database-driven content therefore represents a significant portion of the invisible Web.

Programming can provide higher levels of interactivity with the user. Dynamic content and enhanced interactivity requires the use of various programming technologies that have been adapted to, or specifically created for, Web publishing needs. In addition to making pages more attractive, fun, and interactive, Web programming can handle a myriad of other important tasks including processing data from forms, formatting database queries and their output, automating administrative tasks, setting and utilizing cookies, and validating form input data. In addition, the success of smartphones and tablets has given rise to a huge interest in the scripting and programming skills required to build apps for these devices. These capabilities, along with the possibility of

dynamically generating a new or uniquely configured Web page based on user input or an event, makes learning about programming an important addition to the toolbox of the Web developer.

The starting point for this examination of Web programming is a general discussion of a few programming principles and terminologies that are specific to the Web environment; Schneider and Gersting (2000) provide a good source for introductory programming concepts. This introductory background supports the main agenda of this chapter, which is an examination of interpreted script programming using two popular languages for the Web: JavaScript and the associated JQuery libraries of JavaScript code, and PHP. Web publishers quickly embraced JavaScript and PHP, both of which were developed explicitly for the Web, because they are relatively easy to use and make many website enhancements possible. These scripting languages, when combined with HTML5 and CSS3, can be the basis for many mobile Web apps. This material is not presented from the perspective of a programmer (which I am not), but from the perspective that a basic understanding of these scripting technologies and an awareness of their potential are essential to all those involved in producing Web content and services.

CONCEPTS AND TERMINOLOGY

Programming begins with the idea of an *algorithm*. In simplistic, generic terms, this refers to a series of steps by which a problem can be solved. This definition can be extended to a wide variety of problems and formats describing the steps in the proposed solution. A cookbook recipe, a set of woodworking plans, or driving directions for a trip could all be thought of as describing the steps to solve a defined problem. Programs and scripts are a structured way to solve certain types of problems faced in Web development. In programming, the problem definition and its solution are described in limited, structured terms; the algorithmic solution is expressed in some specific programming language. As with all language, there is a vocabulary and syntax used for the instructions representing an algorithm. Instructions can be grouped together into a *function*. A function, also called a *subroutine* or *procedure*, is a set or block of instructions that execute together to yield a single result; these blocks can be reused in a modular fashion throughout a program. The block of statements associated with the function are enclosed within special punctuation, typically the curly bracket symbols { }. Functions are given names, and they may be predefined in a language or created as needed. Once defined, functions may be invoked later in the script; their action depends on user input or other events, such as a mouse click or a page loading. Information may be passed to a function in the form of a *parameter*, also referred to as an *argument*, when the function is defined. The term *variable* is used in many contexts and can be thought of generically as a placeholder for some value provided or assigned later. In programming, variables refer to named memory locations that store data of various types such as text strings, numbers, or Boolean data. A variable can be assigned a *null* value, meaning no data is stored. Variables are available within limited areas of a script, referred to as its scope, or they can be available for use anywhere, which makes them global in scope. The rules

for referencing variables or declaring them vary with the language used, but all script languages make use of variables.

A distinction is sometimes made between programming and scripting languages. Programming languages such as C++ or Java are written and then *compiled* into machine code; it is the compiled version that is run as an executable file. Scripting languages are not compiled but are interpreted line by line when they are run; they are considered a more lightweight form of programming. So, interpreted programs execute line by line on the fly, while compiled programs are converted to machine code by a compiler, turned into an executable format, and run in that form. To alter or debug compiled programs, the code is corrected and the entire program must be recompiled before it can be tested; interpreted programs can be edited at the line level and run again immediately. Both the JavaScript and PHP languages are examples of interpreted script languages that do not result in compiled programs.

WEB PROGRAMMING

This overview includes only JavaScript and PHP as examples of script languages, but this limited coverage is not intended to imply that they are the only or the best options for many applications; there are other programming languages that are equally important in Web development applications that are well suited to various programming needs. There are also a number of code libraries and many Web service *Application Program Interfaces* (APIs) available to developers that allow for many site enhancements with just a small amount of programming skill. APIs can connect apps to device functions such as the accelerometer, camera, and storage or be used for geolocation and media playback, to name just a few. While not all these topics are covered in depth, a few are highlighted.

An early Web programming development was programming associated with the *Common Gateway Interface* (CGI), which handled many types of information exchange between Web applications and servers, such as processing HTML forms. CGI, as did the GUI Mosaic browser, came out of the NCSA at the University of Illinois at Urbana-Champaign in 1993 (Brenner & Aoki, 1996). CGI was designed mostly for the UNIX environment and it included programs written in many different languages such as C++, PERL, and Python.

Beginning with Windows NT Microsoft has integrated the IIS programs into its OS. The IIS software enables Windows systems to run Web servers and other Internet services on a Windows PC. The IIS suite contains a set of Server Side Includes (SSI), a collection of software that enables many server-side programs, which are programs that run on a Web server when needed. The SSI software is usually associated with servers running Microsoft's IIS services, but these server-side programs are available for other platforms such as UNIX. Microsoft has developed a scripting technology for dynamic content that utilizes SSI software called *Active Server Pages* (ASP). They have embedded scripts that are processed using the SSI programs running on the server. The ASP technology has been coupled with the Microsoft.NET initiative, a broad framework utilizing XML, SOAP, and other technologies to create an Internet-based platform of Web services for Windows systems (Microsoft.NET, 2014). Other proprietary

scripting languages such as Coldfusion (CFM), now owned by Adobe, have also been developed.

Java Server Pages (JSP) and Java servlets are another strategy for creating interactive, dynamic content. Servlets are Java programs designed to run on a Web server and process the instructions contained in JSP pages (Harms, 2001). Because there is potential confusion between Java and JavaScript due to the use of “Java” in both names, their differences should be clarified. Developed by Sun Microsystems in the mid-1990s, Java is a full programming environment similar to C++. Java programming is an object-oriented programming (OOP) language consisting of classes of objects that can inherit properties by virtue of their class membership. Java programming results in compiled applications, applets, or servlets that can run on a JVM, which is available for all platforms. Java is a powerful, general purpose, cross-platform programming language that can be used outside the context of the Web for developing applications (What is Java, 2013).

Java is usually used on the Web in the form of *applets* and *servlets*. Applets refer to programs called within a Web page, retrieved, and then run within the browser by the JVM. However, when Java is used to create programs that run on the server, they are referred to as *servlets*. Servlets require that the Web server either have Java capability built-in to it or that a separate *Servlet container* is installed to be the processing engine for the servlet code (Harms, 2001). Both strategies require knowledge of the Java programming language and result in a compiled program that runs on the JVM.

The main point is that JavaScript and Java are not the same. JavaScript is a lightweight scripting language designed for the Web and intended to enhance HTML. The JavaScript syntax and its object model borrow significantly from the Java language, but they are quite different environments: JavaScript is interpreted by the browser, it does not result in a compiled binary file, and its use of objects is much more limited than Java.

DATABASES AND WEB PROGRAMMING

One of the important motivations for learning Web programming technologies is their application and usefulness in accessing databases via the Web. Riccardi (2003) provides a practical implementation of this, as do a number of other sources. The PHP language discussed later in this chapter is used extensively in Web database development, but it is certainly not the only choice available to the designer. Other options include the Ruby on Rails (RoR) development framework with the Ruby programming language (Ruby on Rails, 2013) or the JSP approach described earlier.

Databases are critical to the activities of most large enterprises and support commercial business transactions and information seeking. The *relational database* type is a common choice for delivery of Web content. There are many options for relational database software applications, including Microsoft Access, Sybase, and Oracle. However, the flexibility of the open-source version of the MySQL database software has made it an attractive alternative for those wanting to do Web development within an open-source framework. Relational databases use a powerful command language called the *Structured Query*

Language (SQL). These queries are often presented within a PHP script and in conjunction with CSS styles, the query output generates a Web page; these are the foundation technologies of a CMS such as Drupal. The general process is examined further in Chapter 11.

APP PROGRAMMING

The large numbers of apps—more than 775,000 for the iPhone alone—are what make smartphones such useful devices (Costello, 2013). However, mobile “apps” can really mean two different things in terms of the underlying programming. Web apps are essentially mobile websites that run within a browser across various platforms; these are implemented for mobile using combinations of HTML5, CSS3, JavaScript, and related technologies. Native apps are more than just scripting and HTML; they are small platform specific programs and are downloadable from one of the app stores such as Apple’s iTunes or Android Market (now Google Play). They are typically written in higher-level languages such as Objective C, C++, or Java. There are advantages and disadvantages to various app development approaches, but clearly more programming skill is required for native app developers. Each environment has Software Developer Kits (SDKs) for each platform that combines development with testing such as Xcode IDE (integrated development environment). There are also open frameworks such as Phonegap (phonegap.com) that only require knowledge of HTML, CSS, and JavaScript that can be combined with various APIs and converted into native apps. Commonly utilized APIs include those used for geolocation or accessing the device camera or storage. Building native apps requires higher-level programming languages that are beyond the scope of this text. For those interested in mobile app development or more in-depth treatment of scripting there are many excellent resources including McFarland (2012), Gardner and Grigsby (2012), LaCounte (2012), and Clark (2012). A few APIs of particular interest to library app developers are highlighted in Chapter 16.

SCRIPT PROGRAMMING BASICS

Scripts are special types of text files that contain commands and instructions that are interpreted by a parsing program or OS. Script files are commonplace on both UNIX and PC systems. In the PC world, simple DOS-based scripts, called batch files, were common and are still used to automate tasks. Shell scripts on UNIX hosts have an equally long history. Other types of scripts include network login scripts and Microsoft’s Visual Basic scripting language that supports macro creation in the Office suite.

Batch files are one of the simplest forms of a script program. Each line in such a script begins with a command, and the parameters (the object of a command) are passed to the commands. The text file that contains the commands is given a **BAT** extension to indicate its executable nature, and the script could be associated with an icon on the Windows desktop. While simple batch files are not as interesting as Web scripts, they illustrate key

concepts of script programs: (1) scripts are simple text files but are recognized as executable by their extension. (2) They are written according to the rules vocabulary of the program expected to interpret them. (3) They can contain comments. (4) They can utilize control structures such as conditional tests. (5) They are interpreted sequentially line-by-line unless there is some intervening control structure.

Microsoft uses Visual Basic script for macros in programs such as PowerPoint and Word. The VB script shown on the left in Figure 10.1 can be used to add a simple interactive macro function to a PowerPoint object, such as presenting a quiz with feedback in the form of alert boxes. The nature of the questions and answers in this example are immaterial; the questions and answers could be about any topic. The point is the VB script controls the response for an incorrect or correct answer, as shown on the right in Figure 10.1. The script uses simple programming techniques to create user interactivity within a PowerPoint slide show by showing an alert box with an appropriate message with feedback when an incorrect response was chosen.

Script languages are very much a command world, so some familiarity with the unforgiving nature of a command line interface is helpful. Command-line environments have strict syntactical rules for command processing and are intolerant of spelling and punctuation errors, as are script processors. A misplaced quotation mark or comma, a mistake in spelling or case, or a mismatched curly bracket, and the script will fail. Another commonality with the command-line syntax is the ability to pass parameters (also called arguments) to a command. A parameter is what the command acts on, identified by its position on the command line. Parameters can be passed to functions as well.

The first Web clients and servers were on UNIX systems, and the people creating these sites usually had extensive experience with shell scripts as well as other programming languages. It is not surprising these early Web developers applied programming techniques to HTML, giving rise to a number of new scripting languages. Microsoft's ASP, JavaScript, and PHP resulted from an interest in enhancing the Web experience as well as extending the functionalities of HTML. Script code blocks can cohabit within HTML or be stored as separate external files. External script files, referred to as

Macros in Visual Basic	Code explained
Sub Wrong()	Declares a subroutine called “Wrong”
MsgBox (“sorry, that’s incorrect”)	A MsgBox function that displays the content shown
End Sub	Ends this subroutine
Sub Right()	Declares a subroutine called “Right”
MsgBox (“Yes—good answer!”)	A MsgBox with feedback for a correct answer
SlideShowWindows(1).View.Next	Advances the slideshow to the next slide
End Sub	Ends this subroutine
Sub Rightlast()	Declares a subroutine called “Rightlast”
MsgBox (“congratulations—you’re done!”)	A MsgBox showing a message for the end of the slideshow to conclude the quiz
End Sub	Ends this subroutine

Figure 10.1 A Visual Basic script used to define three macros in PowerPoint. Each can then be associated with different events.

include files, are recognized by the use of a specific assigned file extension; for instance, JavaScript files end in **JS**. In either case, because Web scripts work within the context of HTML, a firm grasp of both HTML and CSS is required.

Script languages are formal languages that utilize the same types of internal control structures as other programming languages to determine the flow of their execution. Script programming languages are procedural, and the default control structure is sequential control; without some intervening instructions, scripts execute sequentially line by line. Intervening control structures include selection structures, which allow for IF-THEN conditional testing and repetition structures, which allow for looping of a section of code until some condition is false.

Software for Script Programming

Script files are simple text and, as with HTML, they can be created in any text-editing environment. However, many specialized editors are available to make the task of script writing easier. Examples of these include Microsoft's Script Debugger or Notepad++, Mozilla Venkman (one of many Firefox Web developer tool add-ons), or Active State's Komodo Edit. Most simple HTML editors also include various tools to support script authoring and browsers all have developer extensions that can debug and troubleshoot scripts.

Script testing and debugging depends on whether processing is on the client or server (a distinction explored in the next section). JavaScript functionality is integrated into Web browsers and most HTML editing programs, so these scripts are easily tested on a desktop PC without additional specialized software installed, and an Internet connection is not needed. In addition, the browser JavaScript console and many of the Web developer extensions available for most browsers can help debug scripts. Server-side languages, such as PHP, require that a script-processing engine is available along with a Web server for testing; many developers install the LAMP stack (for Linux, Apache, MySQL, and PHP) or the WAMP stack (for Windows, Apache, MySQL, and PHP) to facilitate testing. If these components are installed on the PC testing can be done locally, otherwise testing is done after uploading the script file to a remote server with those capabilities.

Client-Side vs. Server-Side Programming

Web scripts are divided into two different types based on where they are executed. *Client-side* scripts are executed solely on the client host; *server-side* scripts execute on a server with a processing engine or container installed and configured to coordinate with a Web server. JavaScript uses client-side processing and PHP uses server-side processing. Not all client-side script languages work within all browsers. JavaScript was developed by Netscape, and while most browsers can process JavaScript that was not always the case. Visual Basic is a Microsoft technology; IE can process it, but other browsers may not support it.

The distinction between client and server processing extends to Java programs as well, which are designed to run on either the client or the server. Java applets are downloaded with a Web page using the applet tag and are executed by the client JVM. JSP are first preprocessed by some servlet, which is a Java program running in a module on a server. The resulting page is sent to the browser only after this preprocessing that takes place in a servlet container, such as the Apache (formerly Jakarta) Tomcat software. Other server-side technologies previously mentioned are Adobe ColdFusion and Microsoft ASP, both of which utilize scripts within HTML documents that are extracted and processed by the server programs associated with each.

Server-side approaches depend on a successful hand-off by the Web server to some other specific container or processing engine that extracts and processes the code; this process is shown in Figure 10.2. The Web server uses either the file extension or a designated directory location of the requested file to determine that the HTTP file request should be redirected to another program for preprocessing. After the code block has been executed and replaced by the script output, the file is passed back to the Web server to send to the client. Files that end in ASP, PHP, CFM, or JSP all indicate that the Web server program should send the file to another server program first.

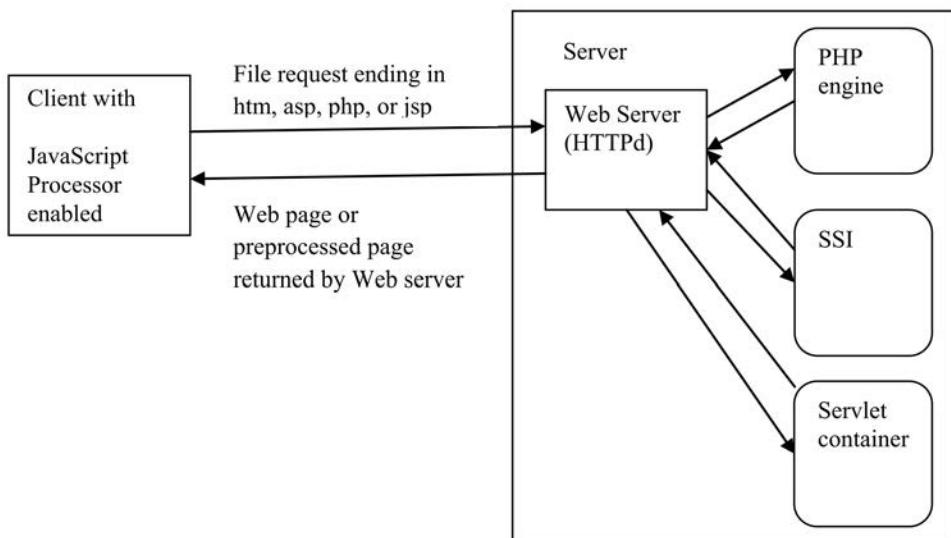


Figure 10.2 A client (browser) may request a static Web page ending with .HTM, or a file that needs additional processing such as those ending with PHP, ASP, or JSP. A file ending in HTM simply is sent to the client via HTTP. However, if the appropriate server-side software is installed, the Web server forwards PHP files to the PHP engine, ASP files to the SSI software, or JSP files to some installed servlet container, where the needed processing takes place. The output of that processing is then returned to the Web server in the same file container that contained the code; this output is then sent to the client for display. If that file happens to contain JavaScript code as well, that must be handled by the JavaScript capabilities of the client.

Pros and Cons of Server vs. Client Execution

Client-side and server-side approaches each have advantages and disadvantages. Client-side scripts accompany the HTML with the result that all the code is easily available by simply viewing the HTML source code, which may or may not be desirable. The easy availability of the script code contributed to the popularity of JavaScript because it can be viewed, copied, and adapted to new uses. Client-side approaches relieve the server of added processing demands and distribute them to the users. However, dependence on the client for processing can be problematic because the script functionality depends completely on the capability of the user agent employed. Standard Web browsers can process JavaScript, but there are mobile and assistive devices that cannot. Server-side approaches are better for accommodating those devices than client-side techniques.

Server-side solutions have advantages. The user only sees the output of the script action, and the script itself can remain private. Server-side approaches also avoid some of the above-mentioned accessibility issues of client-side techniques. However, because they can accept information or parameters from URLs, they can raise security issues (see the discussion of the PHP register globals issues later in this chapter). In addition, these technologies do add to the load of the server, which can create serious performance issues, especially if the script references other server components such as a database. These server load issues can be reduced with the addition of software that enables server caching of dynamic content. Commonly requested Web pages that are generated dynamically from a database can be cached on the server, and the cached copy can be sent in response to future client requests; a new query is performed only if the cached copy is older than a preset timeframe.

Programming Details for Scripts

Scripting languages are all similar in that they share many of the same objectives and functionalities. Although there are differences in the details of how specific tasks are handled, there are common themes and terms employed in all scripting languages. Some of these commonalities are described before examining the examples of specific JavaScript and PHP scripting that follow.

An immediate issue is the textual nature of script programs and the method employed to parse the commands within them. JavaScript and PHP are more rigorous languages than the simple batch files mentioned earlier. Batch files do not require a special instruction terminator other than the CRLF (the “carriage return line feed” ASCII codes) found at the end of each line, and echoed text does not need to be in quotation marks. This is not typically the case with other script languages; instruction terminators are expected to identify the end of the instruction line, and quotation marks must surround text strings. PHP requires the use of the semicolon character for the instruction terminator, although this is not enforced in JavaScript,

it is considered good practice to use it. Both permit either single or double quotation marks to delimit strings. As with HTML, issues arise when text characters with special meaning to the parsing program are used as simple text. Just as character entities in HTML allow the display of characters such as the less than or greater than sign as textual content instead of a tag identifier, scripting languages use “escape sequences” to permit the use of characters such as quotation marks as simple text instead of a delimiter. The backslash character preceding a quotation mark is a way to tell PHP or JavaScript that in that instance the quotation mark is not a delimiter but instead is to be treated as any other character.

In JavaScript, variables are simply declared by providing a qualifying string name set equal to some value; any name may be used, but the usual convention is to use “var” as a keyword first, as in **var some_name = “some_value.”** In PHP, variables can be declared with a name starting with the dollar sign, as in **\$some_name = “some_value.”** Figure 10.3 shows examples of variable declarations with value assignments; the first two are JavaScript, the third is PHP.

Sometimes scripts writers want to add, or *concatenate*, one string or variable with another; this requires an operator (usually “+”) within a script. In addition to the concatenation operation, there are other standard operators used in scripting. The familiar operators from arithmetic and algebra are available and have the same role as special symbols that are associated with some pre-defined action. In addition to the standard arithmetic use of operators (+, -, *, and / for addition, subtraction, multiplication, and division), operators exist for assignment or comparison of values and for Boolean operations (AND, OR, and NOT) as well. The common symbols for each operation are the same in both JavaScript and PHP and are listed in Figure 10.4.

Besides declaring individual variables with a one-to-one name-to-value mapping, it is often useful to reference a list of related name and value pairs, done with an *array*. Like simple variables, arrays have names and refer to memory locations where the array table is stored. Arrays use either a numeric or a name index to determine what part of the list is being referenced. For instance, a numeric key array associated with the days of the week is shown in Figure 10.5. Arrays are explored further in the PHP examples.

Variable naming and assignment of values	Explanation
userID= “Miller”;	In JavaScript, a variable can be simply given a name and assigned a value with the equal sign; the line at left creates a variable named “userID” and assigns it a string value.
var userID= “Miller”;	Here the keyword “var” is used to formally declare this variable; while not required, it is good practice.
\$userID= “Miller”;	In PHP, variable names are preceded with a \$. This line would create the variable “userID” and assign it the string value “Miller”.

Figure 10.3 Examples of creating variable names and assigning them values in JavaScript and PHP.

Type	Operator	Description
Arithmetic	+	Addition
	-	Subtraction
	*	Multiplication
	/	Division
	%	Modulus (division remainder)
	++	Increment
	--	Decrement
Assignment	=	Assign value ($x=y$)
	+=	Assign value as with some added amount ($x+=y$)
	-=	Assign value as minus some amount ($x-=y$)
	=	Assign value as multiplied by some amount ($x=y$)
	/=	Assign value as divided by some amount ($x/=y$)
	%=	Assign value as modulus some amount ($x\% = y$)
Comparison	==	is equal to
	==	is equal to (both <i>value</i> and <i>type</i>)
	!=	is not equal
	>	is greater than
	<	is less than
	>=	is greater than or equal to
	<=	is less than or equal to
Boolean	&&	and
		or
	!	not

Figure 10.4 Script operators used in JavaScript and PHP.

Scripts can respond to *events*, which are actions associated with some function within the script; events cause something to happen. Common trigger events can include a mouse hovering over some area, a click on a button or image, a page loading, or a page being exited. The set of codes associated with these events are *event handlers*.

Key	Value
0	Sunday
1	Monday
2	Tuesday
3	Wednesday
4	Thursday
5	Friday
6	Saturday

Figure 10.5 A numeric key array with corresponding values.

JAVASCRIPT BASICS

This general discussion of structure and terminology can be applied to the two specific languages to be explored further in this chapter: JavaScript and PHP. JavaScript was created in 1995 by Netscape engineer Brendan Eich and released as part of Netscape 2 in 1996 (Shafer, 1996). The name was chosen partly to capitalize on the attention garnered by the Java language developed at

Sun Microsystems, but as discussed earlier, the two languages are quite different. JavaScript is considered a lightweight programming environment designed to enhance HTML. Microsoft developed its own version of this technology called Jscript that was released with IE 3. Because the ECMA (European Computer Manufacturers Association) was the standards body that received and managed the technical specification from Netscape, JavaScript is also known by the name ECMAScript.

JavaScript is an object-oriented language that views a Web document as a hierarchically structured collection of objects, methods, and properties. *Objects* are defined in many programming languages. In broad terms, an object is some defined entity that is named and described with a set of identifiable attributes or properties that are assigned specific values. In the real world, objects are physical things such as a bicycle, which has properties such as color or wheel size that describe it. Script objects have properties but can also be associated with a method, which is a defined set of instructions the object can reference. In addition, parameters can be passed to an object.

When JavaScript was developed, each browser had its own object model, but this object view has been standardized with the W3C DOM (W3C, 2005). Document objects are created using the HTML tags associated with them; further, these objects may have a name attribute defined that identifies a specific object in a particular document. For instance, the tag **<form name = "myform">** defines a form object named **myform**. The JavaScript object model references the W3C DOM standard as well as the other DOM-like features from the earlier Browser Object Model (BOM). There is a subtle distinction between the two: the DOM is a model for a document presented by a browser, while the BOM includes information about the browser itself and defines specific browser-related objects such as the **window** object. JavaScript has a set of core objects that can be referenced in scripts, such as the **document** or **date** objects. JavaScript also permits new objects to be created as needed.

JavaScript uses the *dot notation* for objects and method. This syntax for objects provides object names and identifies associated child objects, properties, or methods. The dot notation convention separates these elements with a period (a “dot”) that reflects the object hierarchy, as in the name **“document.form”** to reference a form object within a document, or **“document.write”** to identify a method that displays information within the document object. This convention is illustrated in some of the JavaScript examples that follow. The browser-centered object hierarchy begins with a window object that contains a document, a history, a location, and references to itself or a parent window. Documents contain elements that can be treated as objects including links, named anchors, paragraphs, forms, and lists. Examples of the components in this object hierarchy are shown in Figure 10.6.

The objects defined in this hierarchy have names, properties, and methods associated with them. The window object has the properties **self** (refers to the current window itself), **frames** (information about each frame window), **default status** (the status area at the bottom of the window that displays link addresses when the cursor hovers over them), or **top** (the window identified as the first opened window in the session). This window object also has methods associated with it, such as presenting a dialog alert box or other prompting dialog box, as well as methods for opening new windows or closing them. Document objects have a large set of properties and methods in JavaScript.

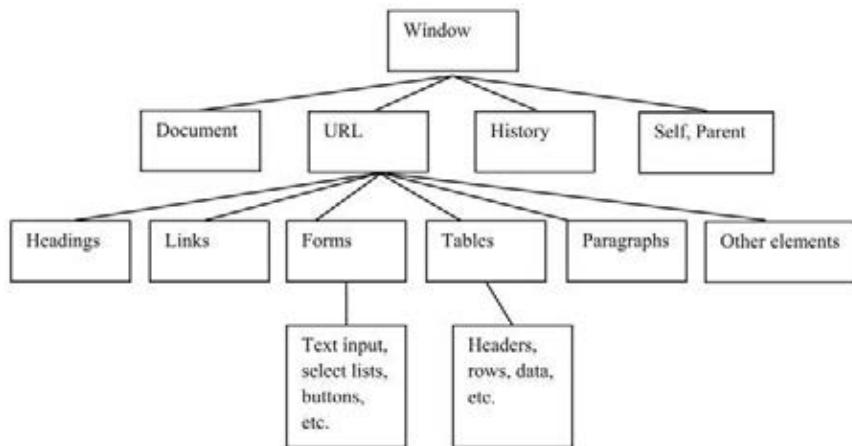


Figure 10.6 Examples of objects in the hierarchical BOM view of Web components beginning with the window object.

JavaScript Examples

JavaScript has been discussed rather abstractly up to this point, but better understanding results from examining examples that demonstrate these ideas. The first JavaScript example is the nearly ubiquitous first-time script everyone learns in scripting: the “Hello World” message that also displays the current date, shown in Figure 10.7.

HTML with JavaScript	Output from a browser display of the HTML file with the script.
<pre><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"> <html> <head> <title>Our first JavaScript</title> </head> <body> <h1>Example 1</h1> <script language="javascript1.2"> <!-- commenting out the script for old browsers document.write("Hello World!" + "&lt;br&gt;"); var today = Date(); document.write(today); // End hiding script --&gt; &lt;/script&gt; &lt;/body&gt; &lt;/html&gt;</pre> </pre>	Example 1 Hello World! Tue Sep 25 13:38:09 2013

Figure 10.7 JavaScript code in an HTML container and its output as copied from a browser display.

This very simple script demonstrates a number of important aspects of JavaScript:

1. JavaScript code blocks can cohabit within a standard HTML file container that ends with the HTM extension. The script is processed by the browser along with the HTML.
2. The script tag pair is used to identify the JavaScript code block; the language attribute identifies this as JavaScript.
3. The script commands are nested within the `<script>` tag container. These commands could have been stored in an external file, called an include file. If this had been done, the source attribute for that external file would be defined (`src = "filename"`) to specify the location of the required script file. The external file would end in **JS** and would contain only the information that would have been placed between the beginning and ending script tags if the script had been placed in the HTML container (i.e., the external file should NOT have any other HTML, such as starting and ending script tags).
4. The first line within the script is an optional HTML comment identifier so that browsers without JavaScript capability will ignore the entire script block. The comment begins with `<!--` which “comments out” that line for systems lacking the JavaScript interpreter; the `-->` ends the HTML comment block.
5. The line of code `document.write ("Hello World!" + "
"` demonstrates an object with a method; the document object can “write” text into the page. It also shows that a parameter, in this case `"Hello World" + "
"` can be passed to this object by its placement within the parentheses. The text to print in the page is in double quotation marks; the plus sign shows the concatenation, or adding, of two text strings. Note that one of the text strings printed in the page is just the HTML for a line break to separate this line of output from the next in the display.
6. The `var today = Date()` declares a variable called “today” and assigns it the value of the JavaScript predefined date object.
7. Finally, the line `document.write (today)` shows that a variable can be referenced in the document.write object.

Another simple script example is shown in Figure 10.8. Note that this script requires an event to trigger the alert function. The output of the script action when viewed in a browser will be an alert box with the message displayed.

This script highlights a few additional points about JavaScript:

1. A function is defined in this script with the line **function ButtonOne()**. It is declared with the keyword “function” and is called “ButtonOne().” Note that the function names end with a set of parentheses, which can be empty or contain a parameter to pass to the function. The block of statements in this function is contained within curly brackets and reference the alert object with the message parameter shown in parentheses.

```

<!DOCTYPE HTML PUBLIC "-//
/W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Example 2</title>
<script language="javascript">
    function ButtonOne()
    {
        alert("Hello World");
    }
</script>
</head>
<body>
<h1>Invoking JavaScript within a HTML
Page</h1>

<input type=button name="buttonJS"
value="click here" OnClick="ButtonOne( )">

</body>
</html>

```

Figure 10.8 JavaScript code for “Hello World” interactive script.

2. Functions are defined in the head area and are called in the body area by some event. In this example, a button was created, and the On-Click event is associated with the function defined in the head area.

One version of a rollover effect with JavaScript is shown in Figure 10.9. This last example in Figure 10.9 highlights these additional points:

1. A conditional test is used (the “***if . . . else***”) to allow for different possible outcomes.
2. JavaScript comments, marked either by // at the beginning of a single comment line or with the start and ending tags /* and */ around a more lengthy comment, are helpful to those who might use or modify your scripts.
3. The object model can be used to do “browser sniffing;” that is, if a browser can work with the DOM model, it means that browser is JavaScript capable. This is more efficient than testing for multiple browsers and versions as was necessary with older JavaScripts.
4. The banner document object is associated with the image by the name attribute value in the image tag.

```

<html>
<head>
<SCRIPT LANGUAGE="Javascript1.2">
<!-- hide script from old browsers
if(document.images) {
/* check to see if browser can handle image objects; better than testing for many browser
versions */
    banner1 = new Image;
    banner2 = new Image;
//makes two image objects

    banner1.src = 'image2mo.gif';
    banner2.src = 'image1.gif';

    //fills objects with specific gifs
}
else {
    banner1 = " ";
    banner2 = " ";
    document.banner = " ";
}
// for old browsers, leave objects empty
//end hiding script -->
</script>
</head>
<body>
<h1>An image rollover</h1>
<a href="file.htm" onmouseover = "document.banner.src=banner1.src"
onmouseout = "document.banner.src=banner2.src">
</a>
</body>
</html>

```

Figure 10.9 JavaScript code for a rollover within an HTML file.

With these examples, along with those you find in other sources, it is important to recognize that there is rarely “one right way” to create a script. For almost any problem, there are many scripting solutions, and there are thousands of examples available on the Web and in books on JavaScript that may address similar functions quite differently. JavaScript has many practical applications, including page redirects, automatic date-last-modified references, image and text effects, slide shows, navigation rollovers, and pop-out menus. While there are annoying and problematic JavaScript applications as well as some known accessibility and security issues, its usefulness in Web development is well established. JavaScript has also found new applications in conjunction with other Web technologies such as XML and CSS, which allow for dynamic styles that exhibit different behaviors based on user interaction. AJAX is used with other APIs to create useful “widgets” in websites such as tools that place location stickpins in a map (Purvis, Sambells, & Turner, 2006). Using the

Google Map API in this manner is an example of a “mashup”; mashups and other Web 2.0 technologies are discussed again in Chapter 16. A good source on Web service APIs and libraries is by Michel (2012).

JQuery and MooTools

Even these simple examples highlight the fact that many people find learning to code difficult and/or tedious. In addition, the tasks the script is to accomplish are often similar to those already handled by other developers in other sites; in fact, JavaScripts have always been routinely recycled by simply copying and modifying the source code from a page where a script has already been written to accomplish a similar task. *JQuery* (jquery.com) is a JavaScript code library that allows a complex script to be added by referencing a library script name and location. Not only does this approach eliminate the need to reinvent the wheel constantly but it also can address browser-processing idiosyncrasies when scripts are run. There are other code libraries to check out, including the Yahoo User Interface Library (<http://yuilibrary.com/>), which also has a CSS library, and the Dojo Toolkit (<http://dojotoolkit.org/>). All these have broad developer communities and scripts that are freely available. A good source for more information about JQuery is McFarland (2012).

Another option for JavaScript developers is the open-source MooTools, described as “a compact, modular, Object-Oriented JavaScript framework designed for the intermediate to advanced JavaScript developer” (<http://mootools.net/>). The developers of MooTools define it as not just a toolkit but as an object oriented and modular framework that is not restricted to the DOM.

PHP OVERVIEW

The *PHP* scripting language grew out of an earlier language developed by Rasmus Lerdorf in 1995 called “personal home page tools.” This was rewritten by Andi Gutmans and Zeev Suraski in 1997 and released as PHP 3.0. The new version of PHP was rechristened with the recursive name “PHP: Hypertext Preprocessor” (History of PHP, 2014). Currently in Version 5.5x, PHP is a well-documented and supported language (see <http://www.php.net/> for the documentation of this language). There are many sources on PHP, including those by Davis and Phillips (2007) and Powers (2006).

PHP is a server-side language designed explicitly for Web development; hence the name “hypertext processor.” It supports interaction with databases including MySQL, Informix, Oracle, Sybase, and the Generic ODBC model. In fact, much of the interest in PHP is because of its suitability and well-established support for database interactions. As an open-source solution, PHP is an especially good fit with the database program MySQL.

PHP Syntax

As with JavaScript, PHP code blocks can cohabit within HTML documents or exist as separate script files. However, one difference from JavaScript is that all files that contain PHP code must end in the PHP extension even if the majority of the file is simple HTML. This is required for the Web server to recognize the needed handoff to the PHP processor. In addition, a PHP engine must be installed on the server to accept this handoff. When the handoff takes place, the code blocks are extracted and replaced with the output of the script; this preprocessed file is what is returned by the server to the requesting client. Because of this difference, PHP code blocks are not seen within the HTML source code that the browser receives. It follows from this that any practical testing of the scripts described here requires access to a Web server with the PHP engine installed. If desired, this environment can be created on your desktop PC by setting up a Web server and the corresponding version of the PHP processor.

Although the look and feel of PHP seems familiar to those using JavaScript, it does have its own rules to learn. PHP can reference objects, functions, and variables, and it has the same operators and data types (strings, integers, floating point, and Boolean) that JavaScript does. As with JavaScript, names are case sensitive, and some names are reserved. It also has a mechanism to “escape” reserved characters; some characters that have a specific meaning to the parser can cause errors when used as simple text within a string, so it is necessary to have “escape” sequences available for those characters. Some key elements of PHP include:

1. PHP code blocks are identified by, and contained within, an identifying tag pair. There are three ways by which such a code block can be identified:
 - a. With a **<php and ?>** tag pair. Note that these are usually on the first and last lines of the code block with all the PHP coding in between (this is the most common and generally preferred method);
 - b. With a simple **<? and ?>** tag pair used as above;
 - c. With the code placed between a **<script language = “PHP”></script>** tag pair.
2. Comments are permitted; single line comments use the double forward slash (//) but may also start with the pound sign (#). Multiple line comments begin with /* and end with */.
3. Each line within the code block requires an **instruction terminator**, which is the semicolon (;) character.
4. Variables can be declared with a dollar sign (\$) immediately followed by a name. Names must start with a letter or underscore character and are case sensitive. PHP allows for different data types with variables, and they do not have to be specified when the variable is declared.
5. Arrays are a special type of variable that can hold a set of values and can be declared with **\$array_name [key] = “some_value”** (see examples in the next section).

6. Functions are declared with the familiar **function name()** where the parentheses can be empty or contain information that is passed to the function; information in the parentheses is called an argument. The block of statements associated with the function is contained within a set of curly brackets { }.

PHP: First Example

With these basics in place, an example is provided of a PHP script that is similar to the earlier JavaScript that displayed a message back into the HTML container. Figure 10.10 shows the code for “**hello.php**.”

This simple code block demonstrates a few of the things that can be done with PHP. Some key observations about this script are:

1. Although the code is embedded in HTML, this file must end with the .php extension; otherwise, it is not handed off to the PHP engine for processing.
2. Because this code block must be preprocessed, the action of the script cannot be previewed on your desktop PC unless a local Web server is installed with the PHP engine. Alternatively, it can be uploaded to some remote host that has both a Web server and the PHP engine installed. After processing, the Web server sends the file to the requesting client. If the source code is viewed in the browser, no PHP code is seen within it; it has been replaced with the output of the script.
3. Each line must end with a semicolon that serves as the instruction terminator.
4. The first line declares a variable called “myName.”

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0  
Transitional//EN">  
<html>  
<head>  
    <title>my php test</title>  
</head>  
<body>  
    <?php  
        $myName = "Joe Miller";  
        echo "<i>Hello world, I'm a PHP script</i>!";  
        echo "<p>My name is $myName";  
    ?>  
    </body>  
</html>
```

Figure 10.10 PHP code in “hello.php.”

5. The next lines contain echo commands. Echo is a common script command used to display some string. Here it simply places the string within the double quotes into the output. HTML code is treated the same as any other text in this string, and it can be “echoed” back into the file, providing markup instructions to the browser.
6. The next echo command contains another text line; it also references the variable declared earlier, displaying the string assigned to this variable in the output file.

PHP Arrays and Functions

The next PHP script file example introduces arrays and functions. Variables can store a single value with a name, but there are times when it is desirable to store multiple values with a single variable name. This is accomplished with an *array*, which is really just a two-column list that has been given a name. Each value assigned is called an *array element*, and each value is referenced by an index entry in the left column of the array. The index can use numbers starting with zero, creating an *enumerated array*. Alternatively, each index value can be given a name referred to as a key, resulting in an *associative array*. For example, consider a set of values such as those shown in Figure 10.11.

Earlier, a function was described as a block of code that performs a specific task that may be predefined or created as needed. Functions are declared with a name followed by parentheses; the parentheses may be empty or hold arguments to pass to the function. Curly brackets are used to house the block of instructions associated with a function. The code shown in “**array-demo.php**” in Figure 10.12 demonstrates both the creation of an array and the use of functions.

The lines with the code `$genre[]` assign values to each key in the array. The use of numbers indicates this is an enumerated array, but string names could have been used instead of numbers resulting in an associative array. As with all variables, declaring them makes them available, but they do not have an action until they are referenced elsewhere in the code. This array is referenced in the count function, which results in the number 5 being displayed as the output. The values in parenthesis are parameters passed to each **foreach** statement; in this case, the instruction is for each successive

value in the array to be associated with a new variable named “**\$temp**” by looping through the array. The statement within the function, shown in curly brackets, displays each value along with the HTML code for a line break, forcing a new line for each element in the output page. One note about the variable **\$temp**: because it is declared within this function, its scope is limited to that part of the script, and it is not available for use outside this function. The distinction between local and global variables is discussed in the last example at the end of this section.

Key	Value
0	Mystery
1	Horror
2	Film Noir
3	Comedy
4	Drama

Figure 10.11 A numeric key array of film genres.

Array-demo.php code	Output of array-demo.php
<pre data-bbox="153 213 744 896"><!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"> <html> <head> <title>test array</title> </head> <body> <h1>A PHP Array</h1> <?php \$genre[0]= "Mystery"; \$genre[1]= "Horror"; \$genre[2]= "Film Noir"; \$genre[3]= "Comedy"; \$genre[4]= "Drama"; echo count (\$genre), "
"; foreach (\$genre as \$temp) { echo "\$temp
"; } ?> </body> </html></pre>	<p data-bbox="744 213 1091 245">A PHP Array</p> <p data-bbox="744 245 1091 278">5</p> <p data-bbox="744 278 1091 311">Mystery</p> <p data-bbox="744 311 1091 343">Horror</p> <p data-bbox="744 343 1091 376">Film Noir</p> <p data-bbox="744 376 1091 408">Comedy</p> <p data-bbox="744 408 1091 441">Drama</p>

Figure 10.12 A PHP code block within an HTML file shown on the left, and the output of the script as it would be presented after preprocessing on the server is shown on the right.

There are many useful predefined functions in PHP. For instance, the array in the previous example could also have been created using the array function with a single line of code as shown in Figure 10.13.

In addition to creating and manipulating arrays, other useful functions include those associated with environment variables. The **GETENV** function has a number of arguments including “**REMOTE_ADDR**” and “**HTTP_USER_AGENT**” that can report on the remote IP address and user agent type and version.

```
$genre = array ("Mystery", "Horror", "Film Noir", "Comedy", "Drama");
```

Figure 10.13 The array function used to create \$genre. Compare this approach to that in Figure 10.12.

PHP: HTML Form Processing Example

This brief examination of PHP concludes with the creation of a PHP script that can accept data from an HTML form using the method POST; see Chapter 5 to review HTTP methods and Chapter 8 for the HTML form code. Specifically, this PHP script can perform arithmetic calculations and report results. Remember, there are many ways to create a PHP script—there are similar examples in books and script library websites, so this code is neither unique nor the only way to accomplish this task (Meloni, 2000). An additional note is that whether the example code works as written on a particular PHP-enabled server depends on how the PHP engine is configured. This version of the script has a problem related to the “register globals” discussion in the next section that may need to be addressed for successful execution.

First, you must have the HTML form that will reference the PHP script in the action attribute of the **<form>** tag. Therefore, two files are needed: the HTML file with the FORM code and the PHP script it references. The HTML code of such a form that will suffice for this example is in Figure 10.14.

The PHP script that accepts the posted data action that is referenced in the action attribute of the form is created next. The form code INPUT tags use

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0  
Transitional//EN">  
<html>  
<head>  
<title>Form that posts data to PHP</title>  
</head>  
<body>  
<h2>A form that can calculate</h2>  
<form method="post" action="calculation.php">  
<p>Value one <input name="value1" type="text" size="10"></p>  
<p>Value two <input name="value2" type="text" size="10"></p>  
Calculation:<br>  
<input name="calculate" type="radio" value="add">Add <br>  
<input name="calculate" type="radio" value="sub">Subtract<br>  
<input name="calculate" type="radio" value="multiply">Multiply<br>  
<input name="calculate" type="radio" value="div">Divide <br><br>  
<input type="submit" name="submit" value="do it!"><br><br>  
<input type="reset" name="reset" value="clear">  
</form>  
</body>  
</html>
```

Figure 10.14 Form code in HTML referencing a PHP script.

NAME attributes that must match the variable names in the PHP script. The source of the data is either the user (for value1 and value2) or a set value for an attribute in the tag, as provided in the radio button input tags. Note again that it is essential that the names used in the form input tags exactly match the variable names in the PHP script. The browser view of such a form is shown in Figure 10.15.

The PHP script code block is in the file calculation.php that is shown in Figure 10.16. Note that there should not be any HTML code preceding the start of this PHP code block. Also, note that there are two different uses of the equal sign; a double equal sign is a comparison operation and a single equal sign is used to assign a value. Figure 10.17 explains the function of different parts of this PHP script.

As noted earlier, this script may not work as written on all servers with PHP available. When scripts fail, the usual suspects are parsing errors caused by omitted semicolons or quotation marks. However, even if the code is reproduced exactly, a change in one of the default settings in the PHP engine setup could result in problems with how the variables are declared in this script—this issue is discussed next.

A screenshot of a web browser window titled "calculation form". The URL in the address bar is "sweb.uky.edu/~jbmill00/calc-form.htm". The page content is as follows:

A form that can calculate

Value one

Value two

Calculation:

Add

Subtract

Multiply

Divide

Figure 10.15 Browser view of the HTML form.

```

<?php
if (($value1 == " ") || ($value2 == " ") || ($calculate == " ")) {
    header ("Location: http://www.uky.edu/~jbmill00/calc-form.htm");
    exit;
}
if ($calculate == "add") {
    $answer = $value1 + $value2;
} else if ($calculate == "sub") {
    $answer = $value1 - $value2;
} else if ($calculate == "multiply") {
    $answer = $value1 * $value2;
} else if ($calculate == "div") {
    $answer = $value1/$value2;
}
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN">
<html>
<head>
    <title>results</title>
</head>
<body>
<h2>Calculation result</h2>
And the answer is: <?php echo "$answer"; ?>
</body>
</html>

```

Figure 10.16 Calculate.phpscript file contents.

Global and Superglobal Variables

Until PHP version 4.2.0, the script as written above would usually work on any PHP-enabled server because up to that version, the setting for “register_globals” was turned on by default. However, with version 4.2 this default setting went to “off” (Using register globals, 2014). This harmless-sounding change had a dramatic impact on a script working with form data like the one just described. The idea of the scope of a variable came up earlier in discussion of the **array-demo.php** script example; a variable defined within a function is local to that function and not available outside of it. However, there are *global variables* in PHP that have unlimited scope and can be referenced anywhere in a script. Further, PHP has **super_global** variables that store environment information in arrays. These are designated with a beginning dollar sign and underscore (**`$_POST`**) such as **`$POST`** for variables coming from

Code	Explanation
<?php if ((\$value1 == "") (\$value2 == "") (\$calculate == "") { header ("Location: http://www.uky.edu/~jbmill00/calc- form.htm"); exit; } if (\$calculate == "add") { \$answer = \$value1 + \$value2; } else if (\$calculate == "sub") { \$answer = \$value1 - \$value2; } else if (\$calculate == "multiply") { \$answer = \$value1 * \$value2; } else if (\$calculate == "div") { \$answer = \$value1/\$value2; }	This starts the PHP code block; no HTML should precede this. This is a conditional test to ensure that the user has provided two values and selected a radio button for the desired operation. If any of the posted values are null, the header function will reload the form page. The symbols are for Boolean OR. If none of those conditions are true, this test is exited and the script continues.
<body> <h2>Calculation result</h2> And the answer is: <?php echo "\$answer"; ?> </body>	This series of if statements are to determine what operation was selected and what should happen for each selection. For instance, if the user chooses the button for Add, the value sent from the form will then match the first string in the series. Since that comparison will be true, the rest of the statement will execute resulting in an answer being stored as "\$answer." This is standard HTML with a second embedded PHP script; its sole job is to present the value now stored in \$answer in the page.

Figure 10.17 The PHP file “calculation.php” and an explanation of the code.

PHP Superglobals	Definition
\$_SERVER	Variables set by the Web server or otherwise directly related to the execution environment of the current script.
\$_GET	Variables provided to the script via URL query string.
\$_POST	Variables provided to the script via HTTP POST.
\$_COOKIE	Variables provided to the script via HTTP cookies.
\$_FILES	Variables provided to the script via HTTP post file uploads.
\$_ENV	Variables provided to the script via the environment.
\$_REQUEST	Variables provided to the script via the GET, POST, and COOKIE input mechanisms, and which therefore cannot be trusted.

Figure 10.18 Superglobal variables from PHP.net.

a form using HTTP POST. Figure 10.18 lists the **superglobals** as documented at the PHP.net site.

With the **register_globals** turned on, PHP scripts could make use of request variables such as those in the calculation example without any prior variable initialization. However, doing so creates potential security issues for PHP scripts. To address this security issue this feature was set to **off** by default with version 4.2, deprecated in version 5.3, and finally removed with PHP 5.4

<pre><?php \$value1 = \$_REQUEST['value1']; \$value2 = \$_REQUEST['value2']; \$calculate = \$_REQUEST['calculate']; ...</pre>	<pre><?php extract(\$_REQUEST); ...</pre>
--	--

Figure 10.19 Two possible solutions to allow the use of the request variables in the calculate.php script for later PHP versions that turned off (or removed) the “register_globals” setting.

(Using register globals, 2014). The previous script must therefore be modified before it can work on PHP-enabled servers unless it is running an older version or the administrator has overridden the default setting.

There are many ways to address this issue in the script example. One solution would be to edit the PHP script and initialize each instance of a variable representing data from the form with **`$POST['variable_name']`**. For instance, **`$value1`** would then become **`$POST['value1']`**. Doing this for every instance of this type of variable can be tedious. Two other more efficient solutions are shown side-by-side in Figure 10.19. At the very beginning of the script, the three variables from the form POST action could be defined as shown on the left of Figure 10.19. Alternatively, another option would be to use the single extract function to import all variables with the superglobal **`$REQUEST`** as shown on the right of Figure 10.19. The **`$REQUEST`** superglobal consists of the combined array of **`$GET`**, **`$POST`**, and **`$COOKIE`**.

These examples just scratch the surface of the many possibilities scripting provides the Web developer. Much of the success of Web 2.0 is due to the many popular widgets that scripts support, such as AJAX-enabled mapping functions.

SUMMARY

Web programming greatly extends the capability of HTML and, when combined with databases, provides the Web publisher with new options for content management. In addition, Web programming is responsible for the success of Web 2.0, mobile-optimized sites, and the huge number of native apps that drive much of the popularity and market success of mobile devices. Such powerful techniques are an important addition to the technology toolbox of all Web publishers and app developers. This is a broad, multifaceted topic that cannot be explored in depth in a single summary chapter, and far more expertise is needed to implement such solutions. Nevertheless, such an overview can at least highlight the wide range of programming solutions and develop an appreciation of the potential of these technologies to enhance and extend the Web experience. Web programming techniques utilize both client-side and server-side technologies, and JavaScript and PHP represent important examples of each approach. The intent of this overview is to serve as a modest starting

point for those interested in further exploration of the area of Web and app programming.

REFERENCES

- Brenner, S. E., & Aoki, E. (1996). *Introduction to CGI/PERL*. New York: M&T Books.
- Clark, Jason A. (2012). *Building mobile library applications*. Chicago, IL: Library and Information Technology Association.
- Costello, Sam. (2013). How many apps are in the iPhone app store. Retrieved May 20, 2013, from <http://ipod.about.com/od/iphonesoftwareterms/qt/apps-in-app-store.htm>.
- Davis, Michele E., & Phillips, Jon A. (2007). *Learning PHP and MySQL* (2nd ed.). Sebastopol, CA: O'Reilly.
- Gardner, Lyza Danger, & Grigsby, Jason. (2012). *Head first mobile web* (1st ed.). Sebastopol, CA: O'Reilly.
- Harms, D. (2001). *JSP, servlets, and MySQL*. New York: M&T Books.
- History of PHP and related projects. (2014). Retrieved January 23, 2014, from <http://us2.php.net/history>.
- La Counte, Scott. (2012). *Going mobile: Developing apps for your library using basic HTML programming*. Chicago, IL: ALA Editions.
- McFarland, David Sawyer. (2011). *JavaScript & jQuery* (2nd ed.). Sebastopol, CA: O'Reilly.
- Meloni, J. C. (2000). *PHP fast & easy web development*. Roseville, CA: Prima Publishing.
- Michel, Jason Paul. (2012). *Web service APIs and libraries*. Chicago, IL: American Library Association.
- Microsoft.NET. (2014). Overview. Retrieved January 23, 2014, from <http://msdn.microsoft.com/en-us/library/zw4w595w.aspx>.
- Powers, D. (2006). *PHP solutions: dynamic Web design made easy*. New York: Springer-Verlag.
- Purvis, M., Sambells, J., & Turner, C. (2006). *Beginning Google maps applications with PHP and AJAX: from novice to professional*. Berkeley, CA: Apress.
- Riccardi, G. (2003). *Database management with Web site development applications*. New York: Addison Wesley.
- Ruby on Rails. (2013). Web development that doesn't hurt. Retrieved September 30, 2013, from <http://www.rubyonrails.org>.
- Schneider, G. M., & Gersting, J. L. (2000). *An invitation to computer science*. Pacific Grove, CA: Brooks/Cole.
- Shafer, D. (1996). *JavaScript & Netscape wizardry*. Scottsdale, AZ: Coriolis Group Books.
- Using register globals. (2014). Retrieved January 23, 2014, from <http://www.php.net/manual/en/security.globals.php>.
- W3C. (2005, January 19). Document object model (DOM). Retrieved January 22, 2014, from <http://www.w3.org/DOM>.
- What is Java technology and why do I need it? (2013). Retrieved May 23, 2013, from http://www.java.com/en/download/faq/whatis_java.xml.

ADDITIONAL READING

- Burd, B. (2001). *JSP: JavaServer pages*. New York: M&T Books.
- Heilmann, C. (2006). *Beginning JavaScript with DOM scripting and AJAX: From novice to professional*. New York: Apress.

WEBSITES OF INTEREST

Coldfusion at <http://www.adobe.com/products/coldfusion>.

JavaScript at <http://www.javascript.com>.

Microsoft.NET at <http://msdn2.microsoft.com/en-us/netframework/default.aspx>.

Microsoft ASP at <http://www.asp.net>.

PHP home at <http://www.php.net>.

Python at <http://www.python.org>.

Tutorials at <http://www.w3schools.com> PHP: HTML Form Processing Example

Web Technologies and Content Management

Early content producers began building the Web with simple text editors and knowledge of HTML. As the Web grew, so did the options for producing and managing content without necessarily working in HTML code that also allowed publishers to dynamically deliver Web content in various forms. Web publishing programs such as Adobe Dreamweaver or Microsoft's discontinued FrontPage provided a WYSIWYG environment where working in code was mostly optional. Microsoft Office incorporated save-as-Web-page-file options to popular programs like Word, making it easy to save work and deliver it in a browser-ready format, but with the result of complex coding in the HTML. All these programs created autonomous, static HTML files that would be uploaded to a server and then accessed by their URLs, and much of the Web is still comprised of such files. However, an increasing number of sites are using strategies designed to allow multiple content experts, often with little technical expertise, to create or update pages they control via a *content management system* (CMS) that delivers dynamically generated Web pages to the user. In this approach, content is updated and maintained in one location and then delivered into various pages using scripts and style sheets.

CONTENT MANAGEMENT SYSTEMS

The term “content management” can be interpreted somewhat broadly. In the most general sense, it can refer to all the various technologies used to collect and manage information for distribution or publication in any medium. This broad meaning can create some confusion about the nature of a CMS because it is used to mean different things in different contexts. There are related and narrower views of content management and the distinctions

among them are primarily functional. The key differences relate to the nature and purpose of the content stored, and the ways it is archived, managed, and delivered. One type of content management is really just simple document management; for example, both Google Drive and Microsoft SharePoint can be used for document sharing and version control. A broader interpretation of content management is its use in conjunction with complex digital repositories and digital asset management systems or course delivery systems such as Blackboard. There are specialized types of CMSs, such as a CCM (Component Content Management) where the primary goal is to store topic content as well as the relationships among those points of content. For example, a CCM would be an appropriate form of content management in organizations that need to support a shift from document-centered writing to topic-level content. Such content could be written in DITA (Darwin Information Typing Architecture), a XML language that can retain a topic-based architecture (Schwartz, n.d.). Another specialized type of CMS is the WCM (Web Content Management) system that has the primary function of managing and delivering website content. The focus of this chapter is primarily on the WCM in the context of preparing information for distribution on the Web in the form of dynamically generated pages that comprise a site, but it concludes with a brief overview of how this topic intersects with the more complex types of content management used to support digital repositories.

In much of the literature and throughout this chapter, the term CMS is used somewhat generically to mean a WCM. The back-end technologies of most CMSs for Web publishing are a database that holds the content and a scripting language such as PHP to handle the database queries and then, in conjunction with style sheets, formats the query output to result in a dynamically generated Web page. Because databases are a central piece of the CMS approach, a brief overview of the relational database model follows.

DATABASES AND DYNAMIC WEBSITES

As discussed in Chapter 10, one of the important motivations for learning about databases and Web programming technologies such as PHP is their application and usefulness in building dynamic Websites. Riccardi (2003) provides a practical implementation of this, as do a number of other sources. The PHP language discussed in Chapter 10 is used extensively with Web database development, but it is certainly not the only choice available to those interested in CMSs. Other options include the Ruby on Rails (RoR) development framework with the Ruby programming language (Ruby on Rails, 2013) or the JSP approach, also mentioned in Chapter 10.

Relational Database Basics

Databases are critical to the activities of most large enterprises and support commercial business transactions as well as information seeking. They are also important in CMS implementations as the back-end data repository, so

a brief review of key database concepts is warranted here. Database systems encompass a large topic and a detailed discussion of them is beyond the scope of this text, but there are many sources devoted to them (Hoffer, Prescott, & McFadden, 2002; Rob & Coronel, 1997).

Databases can be defined as any collection of data and facts that are stored for future retrieval for some process or information need. Computer technology has vastly enhanced the capabilities of database systems. The software that allows users to add, view, modify, or extract information from a database is called a *Database Management System* (DBMS). There are various types of databases, such as flat file databases, where a single table contains all the data; relational databases, where the data and the relationships among the data are stored in a series of interrelated tables; and object-oriented databases, where the data structures are modeled as objects, which can be grouped together to form classes.

The relational database type is a common choice for delivery of Web content. There are many options for relational database software applications, including many proprietary options such Microsoft Access for the PC or SQL Server for the enterprise, Sybase, and Oracle. However, the flexibility of the open-source version of the MySQL database software has made it an attractive alternative for those wanting to do Web development within an open-source framework that also includes the open-source components of Linux, Apache, and PHP in the LAMP package. Relational databases use a powerful command language called the *Structured Query Language* (SQL). In the relational model, data and the relationships among data are stored in database tables; when these tables meet certain criteria they are called *relations*. SQL queries can be written to add or delete records, perform counts or calculations, or extract content from one or more tables to create useful subsets of records. When databases are available via the Web, data entry, query submission, and query results can be done through HTML forms in conjunction with script programming or with a content management editor. When combined with style sheets, the result is a dynamically generated Web page derived from the database query output. A full discussion of the relational model is beyond the scope of this chapter, but it is briefly described later in this section.

Databases play vital roles in all forms of Web commerce by storing customer and product data as well as tracking orders, shipping, and billing. In a CMS, databases are also utilized to create and manage website content. The use of databases, whether for business transactional needs or to enable dynamic Web content where pages are composed “on the fly,” is dependent on a well-designed database, appropriate queries, and the script programming technologies that mediate the overall process. Such databases are the core component of CMSs that allow distributed access to website creation and updates, so designers interested in this approach need to have some understanding of these inter-related technologies.

Early database approaches to delivering website content added a number of formidable complications to site production. To create a CMS in-house, someone had to design, model, and build the database and enter the data. SQL queries had to be written, and a scripting language chosen and utilized to pass those queries to the database system. Static websites only require a Web server to house the HTML pages and to respond to client requests, but

database approaches require the installation of additional server-side software, such as the PHP engine or the Microsoft SSI software. In addition, if the database resides on the same server, performance issues could arise depending on the nature and scale of the site and the query load that result from its use. Caching techniques are often employed to mitigate this issue by storing pages generated by common queries and sending the cached copy instead of querying the database each time a page is needed. This is an efficient way to reduce the load on the database as long as the cached copy is reasonably fresh. The most technically challenging parts of this DIY process were developing the necessary scripts, programs, and style sheets that can take user input, such as a followed link, to create the appropriate query and generate a well-designed response page. If hosted locally, another ongoing challenge is the need to maintain secure and robust servers.

Many CMS solutions manage these database and scripting details for the designer; the database is accessed via graphical views and forms that allow content types to be defined and fields to be added. Content updates are done via a WYSIWYG editor that can reduce or eliminate the need for HTML and CSS skills. The database is still the back-end, but it is managed by the CMS and often never viewed directly by those using the CMS. The open-source MySQL software is a common back-end database in a CMS, and two of the programming solutions frequently used with MySQL are PHP scripting or JSP. There are many books on the details of each strategy, but PHP is so closely aligned with the MySQL database that both these topics are often treated together (Davis & Phillips, 2007; Meloni, 2000; Powers, 2006). The relational database model uses a series of interrelated tables to store data and relationships among data. Each table is designed to store information about a single entity, defined as one of the “things” of interest. An entity might be employees, customers, or products. Entities are described by their attributes, which become the columns of the table; rows within a table are entity instances, or records. One field for each table is designated as a primary key, which is defined as a unique record identifier; this is often numeric, such as a person’s ID number. A table may reference another table’s primary key, which becomes a foreign key in that table. Primary keys and foreign keys create a relationship between the two tables that allows a query to join elements of each table together; these defined relationships among tables allow select queries that can join two or more tables together. The numeric nature of these relationships is expressed in the relational model as its *cardinality*, and these can take the form of one-to-one, one-to-many, or many-to-many relationships.

As with Web designers, designers of databases have tools to help plan and model the database. A common visual diagram is the Entity Relationship Diagram (ERD) that uses symbols to represent the various entities, their attributes, and the relationships among them. Relational databases have strict design requirements, and the process of ensuring compliance with these rules is called *normalization*. The purpose of normalization is to reduce data redundancies and prevent structural errors that can cause anomalies such as unintended data loss when a record is deleted. Normalization takes place in stages, each of which is referred to as a normal form. There are five such “forms” but generally, compliance with the first three is sufficient to prevent

most database problems. The first normal form ensures that each table is about one entity and that a primary key has been assigned, the second normal form focuses on reducing data redundancies and repeating fields, and the third normal form is to eliminate transitive dependences that can occur among fields by ensuring that each field in a table is dependent on the primary key alone.

The power of a well-designed database is its ability to answer questions and report the output. The question must be posed in a specific way; relational database queries use SQL or a graphical equivalent of it. Queries can draw information that meet specified criteria from one or more tables, perform calculations, or join tables together. When used in a CMS, scripts pass these queries to the back-end database; for example, a script can associate Web links with predefined queries or handle queries based on criteria passed to them in HTML forms. This process is transparent to the users of the website who do not need to know anything about SQL. The only tip off to the end-users might be that the URL of a page they are viewing can end with ".php" extension that is appended with a question mark followed by a query reference number; an example is the URL below from the University of Kentucky Libraries site:

http://libraries.uky.edu/page.php?lweb_id=1019

As described earlier, a common type of SQL query is the select query that can draw data from one or more tables. For example, a database table called "students" could be queried as shown below to display first and last names along with email addresses, sorted by last name:

**SELECT Students.FirstName, Students.LastName, Students.Email-
Name FROM Students ORDER BY Students.LastName;**

This SQL statement could then be placed in a script and passed to a database; a script could also accept the database output that along with templates and style sheets result in the presentation of a dynamically produced Web page.

Although an understanding of SQL is useful for those building an in-house CMS or enterprise database, most databases, and CMSs that depend on them, have graphical tools for building such queries. For example, Microsoft Access uses Query by Example (QBE) to graphically identify the tables that are to be included, the fields to show, and any criteria to be applied. Such a query can then be made part of a customized report that includes the desired formatting of the output. A similar process is available in a CMS; for example, Drupal has different views that allow the site manager to select the elements along with any criteria or filters that will determine the content that will be pulled into a page. Databases also have data entry forms to simplify adding or editing records; CMS programs use similar forms for content updates. In many ways, the pages generated by a CMS are analogous to running a database report in Access; a query pulls the data from various tables, applies criteria, sorts, or filters, and then applies formatting to create a professional publication.

WCMS

There are many billions of static HTML files on the Web, and they are the foundation of many sites today. However, creating and managing websites comprised of individual static pages has a number of limitations:

- Content experts often need technical skills to troubleshoot the HTML, upload the files, and manage the hosting site;
- Content and style updates are more time consuming and require multiple steps of editing the file and uploading new versions;
- Version control is problematic as it is difficult to keep track of individual updates to a page or identify all the other pages in a large site that might be affected by that update, resulting in inconsistencies within the site. There also are content redundancies because many pages might present the same components such as contact information in a footer that is repeated in all pages.

CMSs, specifically WCMs, address many of these issues—they can separate the roles of the content expert from that of the Web developer, and they promote efficiency and site consistency by storing information in a database. When the database is then queried, a single update can be pulled into the multiple pages that are derived from it. Choosing a database approach to managing a website involves a significant initial effort, but it can have a significant payoff in long-term ease of maintaining and updating complex sites. A site comprised of numerous static HTML pages requires checking many pages for each minor content update; any pages that are overlooked will continue to provide out-of-date or conflicting information. In a database-driven CMS site, content updates need to occur only in the specific database table that holds that information. These updates can be done with simple Web-based editing software by anyone with access to the CMS database. This shares content responsibility with a large group throughout an organization, but for this approach to be successful, those who can contribute content must be willing to participate and take ownership of that part of the site. Dynamic pages are created by pulling content from the back-end database starting with a request from a Web client, as shown in Figure 11.1.

The engine for the CMS is typically a relational database; the database is updated and queried with scripts, and the output formatted into Web deliverables such as HTML pages using other scripts and CSS style sheets. Underlying the CMS are a Web server and a script-processing engine, such as that for processing PHP. A general view of this model is shown in Figure 11.2. Databases, in conjunction with Web programming and scripting, are central technologies to CMS implementations.

There are many such systems, ranging from in-house solutions built with various modular components, to the popular Drupal CMS (<https://drupal.org/>), a platform that provides all these technologies and features in a single package. A home-grown CMS might use open-source components such as the CKEditor or TinyMCE WYSIWYG editor, the MySQL database, the Smarty PHP template engine, and a link resolver database to manage access to subscription content.

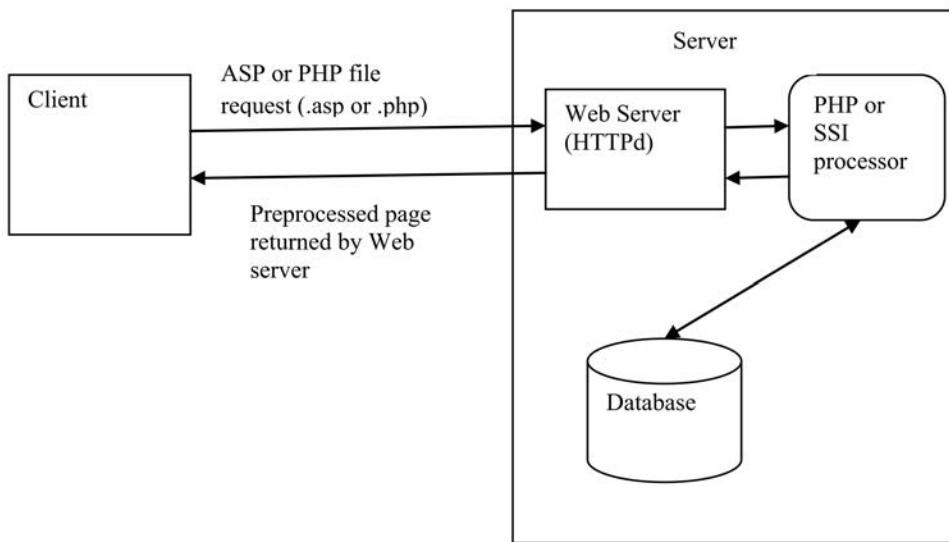


Figure 11.1 A client (browser) requests a file that contains ASP or PHP code. The Web server (HTTPd—d is for “daemon,” i.e., a Web server) recognizes the corresponding file extension and hands the file to the appropriate script processor (the PHP engine or the Microsoft SSI installed on the server). The script is processed there; if the code contains a database query, it is passed to the DBMS, which may be on the same server as shown or on another host. The resulting content is returned to the requesting client by the Web server.

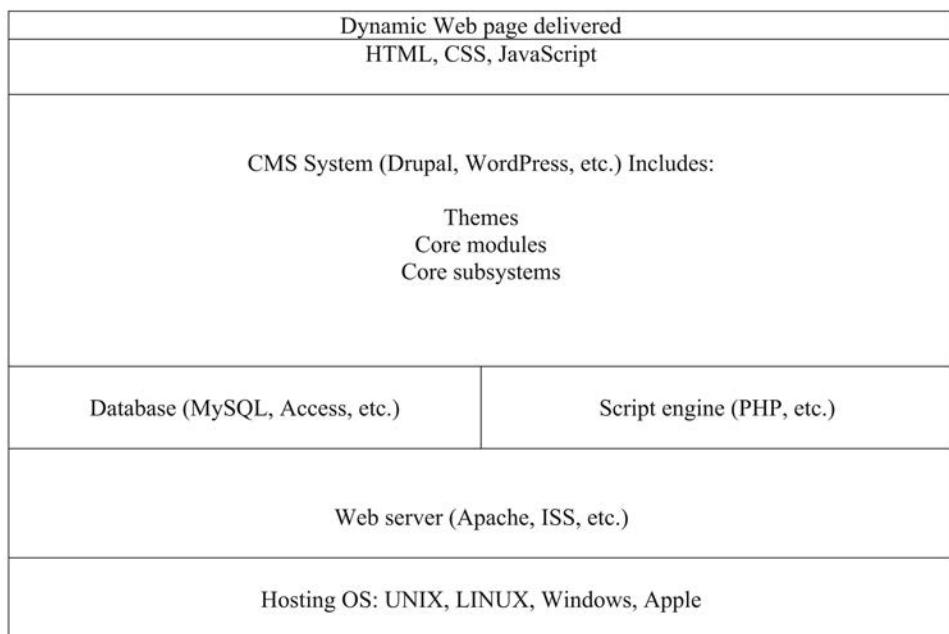


Figure 11.2 Overview of the layers in the CMS model.

CMSs are sometimes also a broader content management framework that includes both site-building tools as well as ways for programmers and developers to customize the various modules that control the options and features available to them (Bryon, Berry, & De Bondt; 2012). These functions are associated with the two main components of a CMS: a content management application (CMA) that allows users to add or modify the site content, and the content delivery application (CDA) that manages these changes and back-end database to update the site. The CMA allows the site owner to define who can access different areas of the site. The CMS software permits a number of user roles such as creator, editor, or collaborator. The site administrator sets up the CMS, chooses the initial modules, and assigns access rights to various users or groups and editors/creators, who then manage updates to assigned pages or create new ones. Editor access is typically restricted to a specific set of files or folders.

CMS Matrix.org lists over 1200 CMS options (Compare content management, 2013), and this discussion will cover only a few of these ranging from those that require significant expertise to establish and manage to very easy to use cloud-based content management systems such as Google Sites or the Drupal-based Library CMS (<http://librarycms.org>).

CMS EXAMPLES

CMSs come in many forms. There are proprietary options, such as Adobe's CQ5 Communique 5 that includes digital asset management, Web content delivery, and social media connections as well as a variety of open-source and hybrid alternatives. There are many options ranging from easy to use cloud-based hosted solutions to those that are locally implemented and managed; some of these are fee based and offer technical support, others are open source and support is primarily from the user community itself. Fee-based pricing varies considerably; the cost might vary from just having ads display on the site to fixed monthly or annual fees. The expertise needed can vary from very little to those that require local expertise in server and database administration as well as in script programming, HTML, and CSS. Almost all local implementations require a standard mix of supporting technologies that include a Web server, PHP engine, and MySQL database. Fortunately, for most OSs, these are available as a package such as LAMP, WAMP, and MAMP, where the first letter refers to Linux, Windows, and Mac OS, respectively, and the "AMP" is for Apache Web server, MySQL database, and PHP. For those with basic Web publishing needs there are a number of free, easy-to-use cloud-based options. Some of these are supported by placing ads on the resulting sites; others such as Google Sites or those sites that are available from an ISP, are a service offered to account holders and customers. These options may have restrictions on file size, traffic, and content but they are often sufficient for a personal page or a small-scale site. Google Sites is a good example of a free Web-hosted CMS solution, but there are many others such as Weebly.com, blogger.com, and WordPress.com, discussed later. A few such as Wix.com support HTML5 templates that are better for creating mobile sites and have "drag and drop" functions that make it easy to modify templates and page elements (Zukerman, 2012).

Google Sites, Weebly, and Wix

Google Sites is a good representative service; it makes it very easy to create and publish Web content with nothing more than a free Google account. They offer dozens of templates for different types of individual or group pages that might be used for family, club, class, or even a small company website. The WYSIWYG makes it easy to add and format content, but with all these editors, it is sometimes necessary to work in the background HTML code to fix some issues. The site URL can be customized and the site shared with the world or just with selected individuals or groups.

Weebly (www.weebly.com) and Wix (www.wix.com) are also popular free Web publishing services that offer premium features for a fee. Both offer many easy-to-use templates, designs, and forms that are suitable for hosted Web publishing needs or blogs.

WordPress

WordPress began primarily as a blog creation site but it has now become one of the main players in the CMS marketplace, accounting for about 56 percent of all CMS-based sites by 2010. This free, open-source blogging and CMS now has over 70 million users (WordPress sites, 2013). As with the other CMSs discussed here, it is based on MySQL databases and PHP programming that is implemented using various plugins, themes, and templates. It grew out of an earlier Web publishing system, the b2/cafelog that began in 2001. A new fork of b2/cafelog was launched as WordPress in 2003 by Matt Mullenweg and Mike Little (Jones & Farrington, 2013). Its roots as a blogging tool were easily extended to its broader use now as a CMS.

WordPress comes in two related, but different forms—a cloud-based version where all server functions are managed for the user or a downloadable, locally implemented version that is more customizable and flexible, but that requires more local expertise. The cloud version is available at WordPress.com and the free software download from Wordpress.org.

As with Drupal, local implementations of WordPress require a Web server, the MySQL database, and PHP version 5.2.4 or higher. Local WordPress implementations are either single site or multisite installs, depending on whether single or multiple domain URLs are needed, multisite implementations add further technical requirements for configuring the Web server.

WordPress has over 24,000 plugins that can be used to extend the content types and administrative functionalities as well as multiple themes to control presentation. Themes are collections of files within WordPress that use PHP scripting to control page blocks and structure, and CSS style sheets to define colors, fonts, floats, and overall appearance. Thus, there could be separate PHP files for page headers, footers, sidebars, or navigation areas. These PHP and CSS files can be used as is or can be edited and customized assuming one has the needed PHP and CSS knowledge to do so.

The hosted solution at WordPress.com is an excellent starting point for exploring WordPress as a CMS; for those interested in a local installation, a helpful guide is by Jones and Farrington (2013).

Drupal

Drupal has become one of the most popular open-source CMS solutions; it is used by many thousands of developers in a wide variety of organizations. Drupal began as a message board in 1999 and it was made available as an online open-source project in 2001 (About Drupal, 2013). As do many open-source programs, Drupal has a large and passionate user community working to extend its capabilities and offer support through user forums. Besides the Drupal site itself, there are a number of sources to learn more about Drupal such as that by Byron, Berry, and De Bondt (2012). The learning curve begins with understanding these Drupal fundamentals and definitions, as described by Peterson (2012):

1. Node: A piece of content.
2. Content type: a standard configuration of a node, such as a blog post.
3. Views: An organized list of nodes.
4. Taxonomy: A list of terms used to tag content.
5. Block: An area of page content, the placement of which is determined by the site template; blocks can be reused throughout the site.
6. Webforms: A form content type used to collect information from visitors.
7. User role: The permissions granted to an account.
8. Module: Contains the programming needed to give the site various functionalities.
9. Themes: Templates chosen for the site that control overall layout and appearance.
10. Input formats: Control the types of data permitted in a field.

In addition to understanding these basic terms and concepts, a Drupal installation has a number of initial requirements that must be in place, including a Web server such as Apache, PHP version 5.2 or higher, and a database server such as MySQL 5.0. Drupal uses *modules* that contain the needed PHP scripting and other functionalities that Drupal can use; other CMS programs may refer to these as plug-ins or extensions (Byron, Berry, & De Bondt, 2012). Some of these are the core modules included with the basic installation; others are contributed by the Drupal community and made available online. The module approach makes Drupal very flexible and extendable—new modules can be added by downloading and extracting them into a folder that can then simply be moved into the modules folder of the Drupal installation. Modules can be combined in various ways to build new custom functionalities for the site. For example, JQuery libraries such as those from Isotope (<http://isotope.metafizzy.co/>) can be used to create a dynamic image gallery (Frazer, 2013). Much of the learning curve for Drupal is related to learning how to select, and then customize, the modules needed from among the thousands that are available.

Two other key components in the Drupal model are users and nodes. Users can be assigned various levels of access through the permissions they are granted. Nodes are the points of content building blocks that are created and maintained in Drupal. Drupal nodes all have some basic properties: an author, creation date, a title, and, of course, content. The default content types included with Drupal are for a basic page and an article, which are quite similar except in the metadata displayed with them, such as author. Other standard content types such as blogs can be easily added, or new customized types can be created. One advantage of a CMS is that content can be reused or repurposed throughout the site. Drupal uses the idea of structural blocks for elements that can then appear on all, or specific, pages. These include items such as headers, footers, menus, and sidebars, and the content of each can be customized as needed. The presentation is controlled by the selection of a theme that provides a consistent layout and look for the site.

All the needed administrative tools are available in a customizable dashboard that provides access to all the key functionalities. The main category links are for content, structure, appearance themes, people and their permission, the modules that are installed, configuration tools, and report generation.

Drupal has become a very successful CMS, hosting an estimated 2 percent of all websites and about 5 percent of those using a CMS (Usage statistics and market share of Drupal, 2013). It is a powerful and flexible Web production solution, but with that comes a certain amount of complexity. In addition, many who use the WYSIWYG editor find it is still sometimes necessary to work in HTML and/or CSS code to achieve the desired result. Web developers using Drupal do not need to learn SQL and PHP, but implementing and managing a local installation has a somewhat significant learning curve of its own. It is not surprising then that various companies and organizations offer and support turnkey cloud-based Drupal solutions. One example mentioned earlier is Library CMS (librarycms.org).

Joomla is another popular open-source option, powering about 3.3 percent of the Web (Usage statistics and market share of Joomla, 2013). Joomla grew out of a development fork of Mambo, an earlier open-source CMS effort. It is based on the same mix of Apache, MySQL, and PHP implemented within an Object Oriented Programming framework (What is Joomla, 2013). It is extendable, with over 9,400 extensions available for download and as with Drupal, has many templates for controlling layout and themes.

LibGuides

Many specialized CMSs are available that can be tailored to meet specific needs. For example, LMSs such as Blackboard are essentially a CMS designed around the specific needs of academic course development and delivery. Libraries have a long history of devoting a significant effort to creating Web-based subject guides to support the research and reference needs of their communities. One of the more popular choices for building such subject guides is Springshare's LibGuides CMS (springshare.com). According to their FAQ:

LibGuides is a hosted system, which means we take care of the servers, infrastructure, data backups etc. so you don't have to worry about any of that. We build a customized system just for your institution and you can

even customize your domain (with optional custom domain mapping). The look and feel of your system is also fully customizable, so you can adjust the stylesheets, and the look to match your institution's branding requirements. (LibGuides FAQ, item 4)

The LibGuides CMS is not just as a wrapper for specific subject guides but is also an option for building an entire CMS library site (Mathews, 2010). The LibGuides CMS interface and feature set make it an attractive CMS platform option for libraries.

Learning Management Systems

Another specialized form of CMS is the LMS that provides a platform for developing and delivering academic courses. The LMS can support courses that are fully online, face-to-face, or hybrid courses with both face-to-face and online components. The LMS can manage all forms of course content, including text documents, video lectures, podcasts, discussions, and social media such as blogs and wikis. Video lectures and discussions may be prerecorded and delivered asynchronously or be streamed live from a class meeting. The LMS typically includes administrative functions for submitting assignments and grading, managing enrollments, tracking student use, and sending alerts.

As with other forms of content management, there are both proprietary and open-source options. The Blackboard LMS (<http://www.blackboard.com/>) is a common proprietary choice estimated to have about half of the LMS market share (Riddell, 2013). It has acquired or merged with WebCT and Angel. Other proprietary choices include Cengage Learning's Mindtap (<http://www.cengage.com/mindtap/>), McGraw-Hill's Connect (<http://connect.customer.mcgraw-hill.com/about/>), Pearson's eCollege (<http://www.ecollege.com/>), and the Canadian company Desire2Learn. Moodle (<http://moodle.com/>) and Canvas from Instructure (<http://www.instructure.com/>) both offer alternatives for those seeking open-source solutions.

As with other complex systems, the LMS must interact with, or connect to other data sources or technologies. For example, a university LMS must connect to the registrar system and pull class registration data or send final grades to it. The LMS also must connect to media servers that contain recorded classes; at my university, many classrooms use the Echo360 system to capture class activities that can then be pulled into the LMS. These added requirements for the LMS also influence the product choice.

DIGITAL REPOSITORIES AND ARCHIVES

As noted at the beginning of this chapter, a general introduction to content management intersects and overlaps with many of the technologies that support various types of digital repositories and digital asset management (Guide to implementing digital asset management systems, 2013). There are hundreds of digital libraries and Open Archive Information Systems (250+ Killer digital

libraries, 2013). These systems are far more complex than the typical WCM, but CMS functionalities may be one of the microservices employed to build the system or deliver its content to users. As with the WCM, there are both open-source and proprietary solutions available to build and support digital repositories, and a few examples of each will be highlighted in this overview. To understand the complexity of these repository systems, a high-level view of the OAIS reference model is necessary.

The OAIS Reference Model

The OAIS (Open Archival Information System) environment has additional requirements and technological overhead when compared to a simple WCM. As with a WCM, it includes content producers and consumers, but it has higher functional requirements beyond simply delivering Web content—it must also manage, archive, and preserve content, while interfacing with content producers, managers, and consumers. The environment model of an OAIS from the Consultative Committee for Space Data Systems (CCSDS) is shown in Figure 11.3 (Reference model for an OAIS, 2012).

Archival repositories are different from a simple CMS-based website in the nature of the content, its dependence on the metadata for both managing the resource and for access to it, and the priority given to preserving and managing an object and a collection. Generic website content is often rather ephemeral, and there is usually little, if any, metadata associated with it. It is also important to emphasize the priority such archival systems give to ensuring the longevity of the content. Early digital initiatives often primarily focused on getting content online as expeditiously as possible in a version optimized for browser viewing on a screen. For example, a scanned photograph that was delivered online as a low-resolution JPEG image where a high-resolution TIFF version might be stored offline, and possibly on media that may become obsolete within a few years. As described later, the OAIS model attempts to address this issue by maintaining both an archival version and a deliverable version of the content that are bound to each other but independent—the deliverable content format can thus change to suit the viewing software or device.

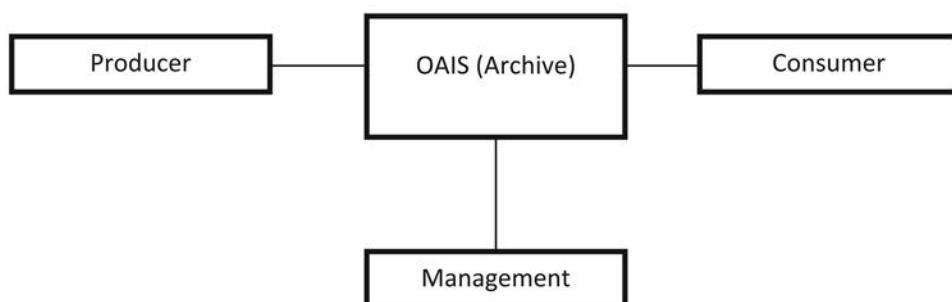


Figure 11.3 The information environment of OAIS archives: content producers, managers, and consumers.

In the OAIS model data objects are interpreted using representational information, which can then yield the information object of interest (Reference model for an OAIS, 2012). An important difference then, is in the nature of the information object itself, which is a package containing the digital object (i.e., bits) as well as object representational information (metadata) and package information (e.g., format, size). In addition to content information, five types of preservation information are applied to a package to provide data about the object's provenance (its source), context (how it relates to other information outside the package), reference (unique identifiers such as an ISBN), fixity (a wrapper to protect it from unauthorized alteration), and access rights (permissions for use). Figure 11.4 shows the relationships among the package and the various types of package information as described by the CCSDS (Reference model for an OAIS, 2012). The goals of a repository system are multifaceted; it must ensure that information is not just preserved, but that it can be discovered and delivered in a way that is readable, understandable, and usable. This is clearly a more complex environment than that of a WCM where the goal is to deliver an information object in the form of a dynamically generated Web page without this added overhead.

There are important variations among these information packages that depend on the functional requirements of archival repositories. The standard packages of interest are the submission package, or SIP (Submission Information Package, not to be confused with "Session Initialization Protocol" defined

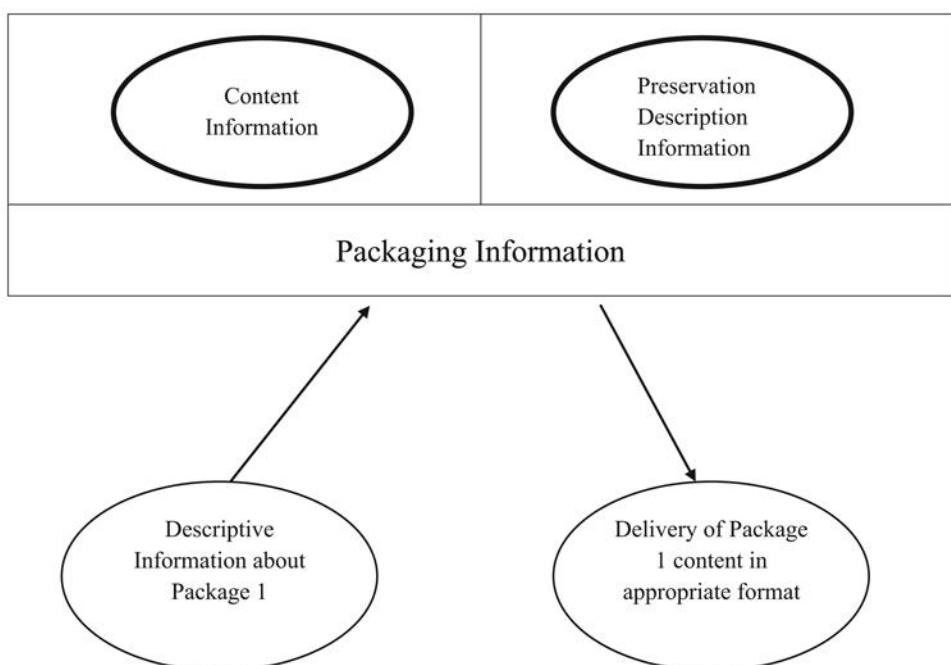


Figure 11.4 An OAIS information package and packaging information that is accessed via descriptive information and delivered in an appropriate format.

in Chapter 5), the archive package, or AIP (Archival Information Package), and the delivery package or DIP (Dissemination Information Package). The SIP is associated with the process of a content producer putting information into the system and the DIP is associated with the process of delivering content to the information consumer. The differing functions of these various packages are supported with different technologies and by different formats; for example, a submission package might be in HTML, the archive package in XML, and the dissemination package in PDF. Although separate, the AIP and DIP are connected and they must be kept current with each other when either is changed or updated, such as with a metadata edit that updates or changes the XML or with a data migration to accommodate a new Web file format. As with CMSs in general, the OAIS model also seeks to limit the need for people to manage all the technical details of the system. One goal of OAIS is to create self-sustaining systems, guided by policies written by the people administering them but implemented as much as possible through programs and coding that automate critical internal processes. For example, such a system can alert administrators to stale content or a data loss, automatically refresh or replicate the content from mirrored sites, or automatically create new deliverables from archived versions.

Microservice Architecture

As with many complex information technologies, there are two general architectural options: a monolithic, layered system that embodies all the needed functionalities within one large-scale solution or a modular approach based on microservice architecture (MSA). The large-scale monolithic approach has been a common choice for many enterprise IT systems. Such systems offer an integrated solution that is efficient and secure, but these monolithic systems are not very flexible or scalable. They also can be somewhat of a “black box” and may involve proprietary inner processes, making it potentially difficult to upgrade or to migrate data to a new system. In addition, dependence on a monolithic solution can be potentially disastrous if the system fails or is no longer supported by a vendor.

MSA is a more flexible, modular strategy. In this model, separate programs or Web APIs such as the Google image viewer handle specific functions. These services then communicate and coordinate with each other to manage the system, deconstructing it into discrete services (Hughes, 2013). Thus, the systems as a whole is not dependent on one vendor, host, or programming language, making it relatively easy for one service to be “unplugged” and replaced independently of any other. As always, there are tradeoffs with this strategy, and disadvantages of the microservice approach include code duplication, the need to integrate code libraries, the necessity of “gluing” various service application containers together, and added configuration and deployment overhead (Morgantini, 2013; Needham, 2012). The services and their integration must also be monitored because unlike layered monolithic systems where a failure in one component is quickly identified by system errors, a microservice failure might not be immediately obvious (Hughes, 2013). These two architectures are not mutually exclusive, and microservices can be used to replace some internal

piece of a large-scale legacy system that has become outdated or that lacks a desired feature.

Digital Repository Examples

There are examples of both the monolithic and microservice approach among the hundreds of digital repositories and archives now available. A few notable examples of “out-of-the-box” solutions include:

- Dspace (www.dspace.org): Dspace is a fully integrated solution for building open digital repositories. It is an integrated solution that includes a Web-based interface for submitting content, item, metadata, and collection management and preservation, and a Web-based retrieval interface to deliver content to end users.
- Digital Commons (www.bepress.com) is a hosted institutional repository software designed for universities, colleges, law schools, and other research centers interested in delivering research articles and other content via the Web. Digital Commons provides the infrastructure and expertise; the institution provides the content. Articles are indexed in the Digital Commons Network index and are also discoverable in search engines or Google Scholar. My university library uses Digital Commons for UKnowledge, a digital collection of scholarship created by University of Kentucky (<http://uknowledge.uky.edu/>).
- ContentDM (<http://www.contentdm.org/>) is another fully implemented single software solution from OCLC that handles the storage, management, and delivery of digital collections to the Web. It includes a digital collection tool to prepare and process data and digital items, a server where data and images can be edited and stored, a Web-based discovery interface, a tool for uploading digital content metadata, and full integration with other OCLC products.

There are open-source options as well, such as ArchivesSpace, described as “the next-generation open-source archives information management application designed by archivists for describing, managing, and providing access to archives, manuscripts, and digital objects” (<http://www.archivespace.org/> overview). Microservice-based approaches are also common; for example, the Kentucky Digital Library (<http://kdl.kyvl.org/>) at my university uses various microservices to submit, manage, and deliver content. The California Digital Library Curation Center (UC3) microservice site (<https://confluence.ucop.edu/display/Curation/Microservices>) lists a number of tools and services to support functions such as content identity, storage, ingestion, fixity, characterization, and access. Their Curation Wiki lists other production services such as the DMPTool for creating a data management plan, EZID for creating and managing unique content identifiers, the Merritt repository service, and their Web Archiving Service. For those interested in additional reading on the

topic of digital repositories and archives a list of websites of interest has been provided.

SUMMARY

CMSs are usually associated with website management but have broader applications, ranging from simple document sharing to large digital repositories. The distinctions among the various implementations of CMSs depend on the nature of the content, its audience and purpose, and the mission and longer range goals of the hosting site. The uses of a CMS for simple document management might just facilitate basic file sharing and version control. An enterprise CMS might require a more granular, topic-based approach best served by a CCM system. Large-scale digital repositories not only utilize a number of CMS technologies but also have other important goals of preserving content and collections as well as improved access through the application of rich metadata. The WCM uses content website management primarily for the ease and consistency of content updates and delivering that content to its intended audience.

Web CMSs have made Web publishing easy and accessible to any individual or organization that desires a Web presence. There are literally hundreds of CMS options ranging from very simple cloud-based options to more flexible and complex locally implemented ones. Choosing the best CMS is dependent on the complexity of the site, the expertise available to manage it, and the cost associated with it. The Web-hosted CMS options make it easy to explore this approach and if organizational needs change, the transition to a more flexible locally hosted version could be considered. CMSs can be content agnostic, but some are optimized to support specific needs such as those for library subject guides or LMSs for course delivery.

Previous chapters in this text focused on building the Web using HTML and CSS coding, and these skills are still very applicable to the CMS environment. Many WYSIWYG editors can result in presentation problems that can be corrected only by working in the HTML view. Although the CMS allows those without these skills to participate, Web developers still need them to implement and manage a site. The success of any CMS approach is ultimately dependent on the level of participation to those given access to it to ensure that the content is being maintained and updated.

The chapter concluded with a brief examination of how digital repositories and OAIS are both similar and different from simple WCM systems. Both are designed to separate some of the technical details from the task of adding or updating content, and both aim to deliver content to users. However, digital repositories and archives have other goals, particularly curation and ensuring that content longevity is independent of the current formats of the Web. In addition, the management of information packages and their subsequent discovery and access by users are dependent on metadata, often in the form of XML-encoded metadata schemes. Thus, XML is an important topic both as Web technology and in its specific application in building and implementing digital repositories. An overview of XML is the topic of the chapter that follows.

REFERENCES

- About Drupal. (2013). Retrieved July 15, 2013, from <https://drupal.org/about>.
- Byron, Angela, Berry, Addison, & De Bondt, Bruno. (2012). *Using Drupal* (2nd ed.). Sebastopol, CA: O'Reilly.
- Compare content management systems. (2013). Retrieved July 8, 2013, from <http://www.cmsmatrix.org/>
- Davis, Michele E., & Phillips, Jon A. (2007). *Learning PHP and MySQL* (2nd ed.). Sebastopol, CA: O'Reilly.
- Frazer, M. (2013, November 11). Building a dynamic image display with Drupal & Isotope. Retrieved November 22, 2013, from <http://acrl.ala.org/techconnect/?p=3739>.
- Guide to implementing digital asset management systems and strategies. (2013). Retrieved November 24, 2013, from <http://searchcontentmanagement.techtarget.com/essentialguide/Guide-to-implementing-digital-asset-management-systems-and-strategies>.
- Hoffer, J.A., Prescott, M. B., & McFadden, F. R. (2002). *Modern database management* (6th ed.). Upper Saddle River, NJ: Prentice Hall.
- Hughes, J. (2013, April 29). Micro service architecture. Retrieved November 24, 2013, from <http://yobriefca.se/blog/2013/04/29/micro-service-architecture/>
- Jones, Kyle M. L. & Farrington, Polly-Alida. (2013). *Learning from libraries that use WordPress: content-management system best practices and case studies*. Chicago, IL: American Library Association.
- LibGuides FAQ. (2013). Retrieved July 7, 2013, from <http://www.springshare.com/libguides/benefits.html>.
- Mathews, Brian. (2010, May 19). Redesigning your website? Why not use LibGuides as your content management system? Retrieved July 12, 2013, from http://theubiquitouslibrarian.typepad.com/the_ubiquitous_librarian/2010/05/redesigning-your-website-why-not-use-libguides-as-your-content-management-system.html.
- Meloni, J.C. (2000). *PHP fast & easy Web development*. Roseville, CA: Prima Publishing.
- Morgantini, D. (2013, August 27). Micro-services. Retrieved December 2, 2013, from <http://davidmorgantini.blogspot.com/2013/08/micro-services-why-shouldnt-you-use.html>.
- Needham, M. (2012, November 29). Micro services: The curse of code duplication. Retrieved November 25, 2013, from <http://architects.dzone.com/articles/micro-services-curse-code>.
- Petersen, Steve. (2012, February 15). 10 Drupal fundamentals. Retrieved July 19, 2013, from <http://blog.thebrickfactory.com/2012/02/10-drupal-fundamentals/>
- Powers, D. (2006). *PHP solutions: Dynamic Web design made easy*. New York: Springer-Verlag.
- Reference model for an open archival information system (OAIS): Recommended practice. (2012, June). Retrieved November 25, 2013, from <http://public.ccsds.org/publications/archive/650x0m2.pdf>.
- Riccardi, G. (2003). *Database management with Web site development applications*. New York: Addison Wesley.
- Riddell, R. (2013, February 6). 12 Learning management system providers and what they bring to classrooms. Retrieved January 20, 2014, from <http://www.educationdive.com/news/12-learning-management-system-providers-and-what-they-bring-to-classrooms/97613/>

- Rob, P., & Coronel, C. (1997). *Database systems: Design, implementation, and management* (3rd ed.). Cambridge, MA: Course Technology.
- Ruby on Rails. (2013). Web development that doesn't hurt. Retrieved September 30, 2013, from <http://www.rubyonrails.org>.
- Schwartz, H. Why CCM is not a CMS: Or why you shouldn't confuse a whale with a fish. Retrieved November 25, 2013 from <http://www.infomanagementcenter.com/members/pdfs/reprints/BP09-12HSchwartz.pdf>.
- 250+ Killer digital libraries and archives. (2013, March 25). Retrieved November 7, 2013, from <http://oedb.org/ilibrarian/250-plus-killer-digital-libraries-and-archives/>
- Usage statistics and market share of Drupal for websites. (2013). Retrieved July 15, 2013, from <http://w3techs.com/technologies/details/cm-drupal/all/all>.
- Usage statistics and market share of Joomla for websites. (2013). Retrieved September 20, 2013, from <http://w3techs.com/technologies/details/cm-joomla/all/all>.
- What is Joomla? (2013). Retrieved July 15, 2013, from <http://www.joomla.org/about-joomla.html>.
- WordPress sites in the world. (2013). Retrieved September 26, 2013, from <http://en.wordpress.com/stats/>
- Zukerman, Erez. (2012, April 9). Create a Website easily with Wix (even the free version). Retrieved July 11, 2013, from <http://www.pcworld.com/article/253379/wix.html>.

WEBSITES OF INTEREST FOR CONTENT MANAGEMENT

- CKEditor at <http://ckeditor.com/>
- TinyMCE at <http://www.tinymce.com/>
- Smarty at <http://www.smarty.net/>
- PHP at <http://us.php.net/>
- MySQL at <http://www.mysql.com/>
- Drupal at <https://drupal.org/>
- Joomla at <http://www.joomla.org/>
- LibGuides at <http://www.springshare.com/>
- WordPress at <http://wordpress.org/>
- Merritt Repository Service at <http://www.dataone.org/software-tools/merritt-repository-service>; California Digital Library Curation Center: <https://confluence.ucop.edu/display/Curation/Home> and <http://www.cdlib.org/services/uc3/>
- EPrints at <http://www.eprints.org/us/>
- AtoM open source archival description software at <https://www.ica-atom.org/>
- ArchivesSpace at <http://www.archivesspace.org/>
- Archivists Toolkit at <http://www.archiviststoolkit.org/>
- Omeka at <http://omeka.org/>
- HUBzero open source Web publishing platform at <http://hubzero.org/>

This page intentionally left blank

XML Primer

Internet content comes in a multitude of formats representing the variety of digital documents found on the Web. These formats can be divided into those that use proprietary binary file types and those that use open text-based standards. Proprietary formats have many advantages and a long history that predates the Web. By combining the content with control codes for presentation and structure, binary formats are a highly efficient way to encode and present documents. However, this efficiency comes at a cost of exchangeability because users must typically have the software that created them to use or view the content. Other approaches that use simple text for content as well as codes to control presentational directives also have a long history in the digital world. Given their simplicity and portability, it is not surprising that Web founder Tim Berners-Lee used a text format to develop HTML, which has become the pre-dominant form of Web content.

There are significant limitations to HTML, especially its inability to provide much information about what the content being marked up actually *means*. For instance, the markup tells you and a browser that the content in a list are list items, but provides no information about what the nature of those items in the list actually is—it could be a grocery list or a list of people in a department. Traditionally, data structuring that provides metadata in the form of field definitions has been accomplished through the creation of a database resulting in a proprietary binary format. XML does for data structuring what HTML does for document presentation by providing an open text-based alternative to structure and exchange data. Since its introduction, a number of XML technologies have emerged to become part of today's Web, including RSS, SVG, SOAP, and WML.

XML overcomes the limited document structure of HTML, which is almost exclusively focused on presentation. HTML5 introduces a few semantic elements,

but as with all HTML, it has a predefined set of elements. XML tags are extensible, that is, elements are not predefined and can be created as needed. XML allows the creation of self-describing documents with markup that conveys information about the content the tags contain. Although XML is a useful method to structure data and for data exchange, it is not the only option available to programmers. There are situations where a full XML implementation might be best, but alternatives such as JSON (JavaScript Object Notation) and YAML (for “YAML Ain’t Markup Language”) might be a more appropriate and simpler way to accommodate data exchange in some situations. These alternatives will be briefly examined at the end of this chapter.

XML was often equated with the Semantic Web but this is somewhat of an oversimplification. Early on, the Semantic Web initiative was described on the W3C site as “a common framework that allows data to be shared and reused across application, enterprise, and community boundaries” (W3C, 2001). It is important to note that the term “semantic Web” has come to be used in different ways. The original W3C initiative is sometimes referred to as the “*Big-S Semantic Web*.” In this context, it refers specifically to an XML-based Web utilizing the W3C data formats of RDF and OWL (W3C, 2013). However, the term “semantic Web” can also be used more generically to describe a Web enhanced by many forms of metadata and the various technologies that can make use of more granular data connections, which in turn can support different implementations of semantic search. There is then a distinction between “Semantic Web” as a formal name that refers to a particular mix of standards and the broader notion of a “semantic Web” that could include many possible ways to add connections and meaning to data. As the Web has evolved, so too has the W3C’s view of what “a web of data” means and how it can be achieved:

The Semantic Web is about two things. It is about common formats for integration and combination of data drawn from diverse sources, where on the original Web mainly concentrated on the interchange of documents. It is also about language for recording how the data relates to real world objects. That allows a person, or a machine, to start off in one database, and then move through an unending set of databases which are connected not by wires but by being about the same thing. (W3C, 2013)

Web founder Tim Berners-Lee was an early promoter of the idea of the Semantic Web as the ultimate future Web, referred to by some as Web 3.0. Berners-Lee asserts an XML data-centered Web will enable direct data exchange among applications and enhance intellectual access to content through search engines designed to exploit its structure (Berners-Lee, 1999; Berners-Lee, Hendler, & Lassila, 2001). Although this vision would utilize XML, it also depends on other technologies and standards. Standards for the original description of the Semantic Web include the Resource Description Framework (RDF), which supports metadata exchange and interoperability, and OWL (Web Ontology Language), designed to facilitate content processing by applications. Both RDF and OWL are documented at W3C. Although the details of this broader application of XML and interrelated Semantic Web topics are beyond the scope of this chapter, some of these topics are revisited in the discussion of semantic search in Chapter 15.

This introduction to XML has four major goals: (1) to introduce the nature of XML documents and the related ideas of well-formedness and validity; (2) to explore the codification of XML with DTDs or XML Schema Definitions (XSD); (3) to consider how XML documents can be presented and transformed with styles; and (4) to examine a few alternatives to XML as well as examples of XML implementations such as RDF and RDFa that support metadata exchange and semantic search. Many books and tutorials on XML provide in-depth coverage of these topics; an excellent short introduction to XML is by Yott (2005).

XML: THE EXTENSIBLE MARKUP LANGUAGE

XML, like HTML, was derived from the SGML, which was developed to accommodate markup of digital data. SGML is not a specific markup language but a meta-language used as a standard for creating specific markup languages. The SGML standard evolved from IBM's GML and it became an ISO standard in 1986. SGML is a very broad standard developed to accommodate multiple forms of digital publishing. SGML is not particularly suited to the more focused needs of Web developers. XML was first proposed at the SGML 96 Conference as a Web-centered, lightweight version of SGML. The goal was to create a dialect of SGML suited to Web processing that is similar to that of HTML, while retaining interoperability with those preexisting Web standards. In essence, XML is a "scaled down" subset of SGML optimized for the Web. Like SGML, XML is a meta-language and it can be used to create many specific markup language implementations.

As with SGML, XML is a text-based standard, utilizing tags in ways that are similar to HTML. However, there are important differences. HTML tags are predefined within the HTML standard. In XML, potential tags are unlimited and self-defining; hence, the markup is "extensible." XML is supported in two current versions; as of 2013 version 1.0 is in its fifth edition, and version 1.1 in its second (W3C, 2013). Version 1.1 is not substantively different from the previous version except for its support of extensions to the Unicode 2.0 character sets (Bray et al., 2006).

XML VS. HTML

This examination of XML begins by identifying its commonalities and differences with HTML. On a superficial level, HTML and XML look quite similar: Both are text based, and both define elements with tags that provide information to a parsing program. Each uses the less than (<) and greater than (>) symbols to identify tags; tags can have attributes that further refine them. However, there are some key differences: HTML uses a predefined set of tags and attributes, does not permit the creation of new tags as needed, and is designed to simply display or present data. In contrast, XML tags are unlimited and designed to structure data by describing what the content means; XML markup is thus semantic and similar in function to that of a database. HTML creates some structure to the content, but only at the level of identifying the logical parts of the document, making HTML elements such as titles

and headers distinguishable from other parts such as paragraph text. HTML5 introduces new semantic elements for documents that provide alternatives to generic DIV containers for common features such as headers, footers, article, and navigation areas, but these do not provide metadata related to the content of those containers. A search engine performing term weighting can use HTML information and text decoration, but the markup does not inform the program about the nature of the content itself.

Although XML documents are just text, they are not intended to be read that way by people. Programs parse the XML data and use styles or other transformation technologies that produce output for presentation. XML's strength is that it can accommodate data exchange between programs and transmit information across multiple platforms. Although there is considerable interest in its application to one vision for a future Web 3.0, XML is not intended to displace HTML. There is a huge volume of information in HTML format that will remain on the Web. In addition, HTML will likely remain a common output format for XML documents because Web browsers cannot directly process XML content. This is not surprising as browsers were designed specifically for HTML; they only show the tag coding of XML documents with hierarchical nesting unless additional presentational style information is provided. XML is more powerful and flexible than HTML, but it is also a less forgiving environment for the creator. Much current Web content created by nonexperts is not well-formed HTML; tag omissions and coding errors are commonplace. The forgiving nature of the HTML parser (the browser) tolerates many of these errors and usually presents the document in spite of coding errors. In contrast, XML documents must be well formed; any code violations cause parsing errors that halt document processing.

XML KEY CONCEPTS

Even though XML is a subset of the broader SGML standard, it is still a large and complex standard. Some of the key summary points about this standard are:

- XML is a simplified subset of the SGML meta-language optimized for the Web.
- XML is for structuring data as opposed to determining how data will be displayed or presented. XML contains rules for using text formats to define data structures unambiguously in new markup languages.
- XML looks like HTML; it is hierarchical and uses tags to identify elements. Tags are identified with the less than and greater than symbols (< >), and they can have attributes in the form of name/value pairs. In HTML, the tag set is predefined; in XML, the tag set is extensible. XML tags are delimiters and what they mean depends on the program using it.
- XML is not intended to be a replacement for HTML.
- XML code can be viewed as text, but it is intended to be processed by some program.

- XML is “verbose by design”; that is, much of the file size will be associated with the markup text overhead that is responsible for the data structuring.
- XML is really a family of interrelated technologies, including XLINK (how links are made in XML), CSS, XSDL, XSL (Extensible Stylesheet Language), and XSLT (XLS Transformations).

Key Components of the XML Specification

The full XML specification, maintained by W3C, includes these key components:

- XML allows for the insertion of comments. Comments allow the insertion of information that is for human consumption but ignored by the parser. Just as in HTML, comment declarations are identified by the less than sign, exclamation, and two dashes and end with a double dash and the greater than sign, as in: **<!--this is a comment-->**.
- XML allows for entity and character references. Text-based markup languages have parsing problems when certain reserved characters are needed to be simply treated as text. HTML has a long list of character entities; XML uses only a small subset of five of these for the following characters: ampersands, less than signs, greater than signs, apostrophes, and quotation marks. In addition to these special character entities, XML allows for the creation of a named entity to represent a string of characters that may occur frequently in certain documents such as a copyright statement or trademark. As in HTML, character and entity references begin with the ampersand character and end with a semicolon.
- XML allows for the insertion of processing instructions. These instructions are identified by the use of the less than sign followed by a question mark (<?) and contain information for the parsing program. Processing instructions are not part of the XML content and are passed directly to the XML parser.
- PCDATA: This stands for “parsed character data” and is the information or content itself that is being structured by the markup. Anything that is not markup is parsed character data.
- XML permits empty elements. As in HTML, there is markup that does not apply to any content. There are empty elements in HTML such as the image, line break, and horizontal rule tags (``, `
`, `<hr>`) all of which simply identify something that is supposed to display at that location; there is no content between the beginning and end tags of empty elements.
- XML includes the possibility of a DTD. This is the Document Type Definition, and XML languages were often designed to reference

a DTD to model that specific markup language. The DTD contains all the tags, attributes, and rules for the markup language. A DTD is not required, so XML documents can be DTD-less, or alternatively, the strategy of creating an XSD may serve the same role as a DTD.

XML and Data Structures

As noted earlier, database programs have traditionally been the primary tool for structuring data. Database programs result in files that are highly efficient containers for data structuring, but typically, they utilize proprietary binary formats that require users to have the same program or software that facilitates data exchange. Data exchange among various programs is thus limited and depends on software and standards such as the Open Database Connectivity model. In contrast, XML has the advantage of being an open text-based standard not dependent on a specific program. While XML can structure data, it is not intended to displace database systems but to complement them. For instance, XML can serve as a data exchange format that retains much more meta-information than other common text-based exchange formats such as **CSV** (comma-separated value) files (Graves, 2002). The process of creating a database can be lengthy and includes systems analysis, modeling techniques, data dictionary creation, and prototype development. XML solutions often require a similar investment in the creation of a document model expressed in a DTD or XSD; these document-modeling languages are discussed later in this chapter.

XML key concepts can be summarized as follows: (1) XML is a meta-language that supports an unlimited number of document types. (2) XML documents are

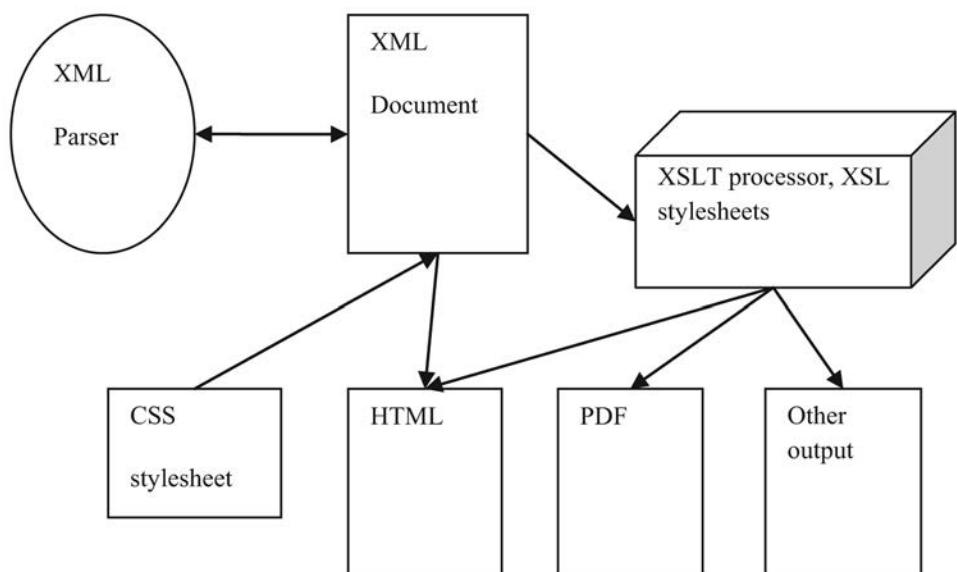


Figure 12.1 XML is designed to interact with many related technologies. The parser checks for well formedness and validity; CSS can yield HTML views; XSLT transformations can process the XML into other forms

intended to be processed by programs that can make use of the data structures built into the markup. (3) XML vocabulary and rules may be codified in a DTD or XML Schema. (4) XML depends on a number of related technologies to enable data exchange and presentation of content for human consumption, as shown in Figure 12.1.

Creating XML Documents

Any text editor can be used to create and edit XML documents. However, a word of caution about character sets used by simple text editors—when XML validation is attempted, if the encoding type used by the editor does not match the default or specified encoding directive to the validating program, errors occur. For instance, quotation marks may appear fine but if they come from a different character set parsing errors will result. Most developers use XML-aware editors such as EditX and Microsoft XML Notepad. These programs have features such as tag auto-complete and attribute name support; some can also validate the XML.

An examination of examples of XML documents is informative before continuing with a more detailed analysis of XML. A sample XML document is referred to as an *instance document*, a single instance of some class of defined document type. While the term *document* often carries the connotation that the content is literally a narrative document such as a Web page or an article, this may or may not be the case in the context of XML. Here *document* is meant to refer to any XML container, whether or not it is literally a document in the sense a Web page is.

XML-based book catalog	XML-based directory of student information:
<pre><?xml version=“1.0” encoding=“UTF-8”?> <!DOCTYPE catalog SYSTEM “cat.dtd”> <catalog> <book binding=“soft” ISBN=“0-1111-2222-3”> <author alive=“no”>P.G. Wodehouse</author> <title>The Mating Season</title> <date>1949</date> <publisher>H. Jenkins</publisher> <city>London</city> </book> </catalog></pre>	<pre><?xml version=“1.0” encoding=“UTF-8” standalone=“yes”?> <directory> <student gender=“female”> <first_name>Mary</first_name> <middle_name>Jane</middle_name> <last_name>Smith</last_name> <address> <street>14 Oak Ave</street> <city>Middletown</city> <state>NJ</state> <zip>11125</zip> </address> <phone>777-968-1000</phone> <email>mjsmith@mail.com</email> </student> </directory></pre>

Figure 12.2 Example of two possible XML documents.

Two examples illustrate the structure of XML. In the first example, XML is used to create a catalog of bibliographic information for books, similar to the type of information that would be found in a library database; in the second, it is used to create a student directory. These hypothetical types of XML documents could look like those shown in Figure 12.2.

These XML documents begin with an *XML declaration*, which looks like a processing instruction because of the question mark in the tag but technically is not (Harold & Means, 2002). This statement is not part of the XML document but provides the XML version information as well as other attributes, such as the text encoding being referenced and whether the document is standalone or not. In these examples, the XML version is 1.0, and both declarations specify the encoding to be UTF 8, which is 8-bit ASCII. The document on the right of Figure 12.2 also has the attribute and value **standalone = “yes”** to indicate that this is to be processed based on the generic rules of XML without a reference to a specific DTD or XSD.

XML documents use tags to identify the elements and must begin with a *root or document element tag*. This is the top-level tag, which by definition has no parent. All XML documents must have a starting tag not nested within any other tag. XML documents have a hierarchical structure, and tags can be nested within tags. This tree-like structure is also referred to as a *node tree*, and each element is a *node*. Note that tag names are case sensitive, and all tags must have end tags, including tags for empty elements.

The XML specification allows attributes to be defined to describe a property of the element. The **<book>** tag in the first example has two attributes named “**binding**” and “**ISBN**.” The set of defined element attributes are determined much as they are in the logical modeling phase of database design; they reflect the essential descriptive characteristics of an entity. How attributes are defined and how they are used varies in different XML languages. For instance, the designer has the choice of adding information as an attribute of an element or as a child element. In the example document on the right in Figure 12.2, “**gender**” could be defined as a separate child element of **student**, as with the child tag pair **<gender>female</gender>**, shown here with the PCDATA “female,” or it could be defined as an attribute within the element **student**. The choice is dictated by the purpose of the information and how it will be used. Generally, data are stored as elements, and information about data (metadata) is stored as attributes.

Well-Formedness and Validity

XML requires that documents be *well-formed*. This means they must comply with the basic rules of XML as defined in the W3C standard. The key requirements are the presence of a root element; matching tag name pairs (the match must be exact in terms of upper and lowercase use); closing tags for all tags including empty elements; and proper tag nesting within tags. An instance document that complies with these rules is considered well formed.

To be *valid*, the instance document must reference some DTD or XSD and comply with it. DTD and XSDs represent two ways to define a specific markup language and these methods are compared in the next sections. For now, the main idea is that *validity* is a separate test beyond well-formedness. Validity

requires verifying the document against a specific set of documented rules. XML permits the creation of documents without reference to a specific DTD or XSD, so a document can be well formed but not valid. Formalizing the rules of a particular XML explicitly defines the language in a way that is useful to both parsing programs and to the people who use it; such a formal definition is called a schema. There are two main approaches to this process: the original DTD schema language derived from SGML and the newer XSDL. DTDs are considered first.

THE DOCUMENT TYPE DEFINITION (DTD)

The “**catalog**” XML instance document example in Figure 12.2 references an external DTD, but the “**directory**” XML instance document does not. Simple XML documents can be created without going to the trouble of documenting all the tags and rules formally in a DTD, and such an XML document is said to be “DTD-less.” The decision to create DTD or not depends on the complexity of the markup language, the number of documents it will have to support, the number of people who might wish to create such documents, and the requirements of the programs developed to process them. There are advantages to precisely defining the tags, attributes, entities, and relationships of a markup language, and the DTD is one way to do this. However, writing a DTD is a complex task that involves learning a separate DTD language. Specifically, DTDs:

- Define and name all the entities and elements available in a document.
- Define the order in which elements will appear, their data type, and whether or not they are required.
- Determine the numeric constraints for the occurrences of elements by specifying whether they may occur not at all, once only, or multiple times.
- Define the nesting rules by showing child element lists for each element.
- Define all the attributes of each element, their data type, and any constraints on attribute values.

The use of a DTD for documenting XML tags, attributes, entities, and nesting rules was inherited from the SGML standard. DTD construction uses a specific syntax to make declarations, which adds to the learning curve for those using this approach to document a markup language. The familiar HTML standard was derived from SGML and uses a DTD through version 4.01; in fact, it has three DTDs: strict, transitional, and frameset. These are built-in to the browser, which serves as the HTML DTD processor.

A DTD can be internal, which means it is included within the instance document itself, or it can be external to the document, in which case its name and location must be specified. In addition, an XML document can have both internal declarations and an external DTD. The Document Type Declaration

contains the information about the DTD that is in use and specifies its location. Declarations always begin with a less than sign and exclamation point. If the DTD is external, the statement will be of the form:

```
<!DOCTYPE rootname SYSTEM "some.dtd">
```

In this declaration, **rootname** is a placeholder for the actual name for a root element in a particular XML implementation, and the location and name of the DTD file are provided in quotation marks preceded by the word SYSTEM in capital letters. If the DTD was internal, the DTD itself would follow a left bracket character within the DOCTYPE tag and end with a right bracket and greater than sign as will be discussed further in the next section. Such an internal DTD is immediately followed by the XML instance document as summarized in the code here:

```
<!DOCTYPE rootname [  
    DTD content follows here . . .  
]>
```

The XML document instance would then follow this internal DTD.

DTD Elements

To create an element, a declaration is made in the DTD that provides the element name as well as any child elements it can contain. In addition, special characters are used to indicate whether these nested elements occur zero or one time (?), zero to many times (*), or one to many times (+). The contents of the

1. <!ENTITY NIP “not in print”>
2. <!ELEMENT catalog (book)+>
3. <!ELEMENT book (author*, title*, date*, publisher*, city*)>
4. <!ATTLIST book
 - a. binding CDATA #IMPLIED
 - b. ISBN CDATA #REQUIRED>
5. <!ELEMENT author (#PCDATA)*>
6. <!ATTLIST author alive (yes | no) #IMPLIED>
7. <!ELEMENT title (#PCDATA)*>
8. <!ELEMENT date (#PCDATA)*>
9. <!ELEMENT publisher (#PCDATA)*>
10. <!ELEMENT city (#PCDATA*)>

Figure 12.3 A simple DTD for the book catalog; lines are numbered to match discussion in text.

external DTD file “**cat.dtd**” shown in Figure 12.3 models the structure of the “**catalog**” document example from Figure 12.2.

Line 1 of this DTD defines an entity reference that allows the shortcut “NIP” to be used for the string “not in print” in these documents. Line 2 defines the root element **catalog** that contains the child element **book**, which must occur at least once, indicated by the plus sign after the element name. Line 3 declares that the element **book** has five child elements, listed in parentheses; the asterisk character means all of these may occur zero to many times within the element **book**. Line 4 defines the attributes **binding** and **ISBN** for the **book** element to be character data; **#implied** means an attribute does not have to be present, but **#required** attributes must appear with the element. Line 5 defines the **author** element as parsed character data (#PCDATA). Line 6 defines the attribute “alive”; this attribute is permitted to have only the value yes or no as indicated by the vertical pipe character, which is for Boolean OR.

The document declaration in the “**catalog**” XML document example shown on the left of Figure 12.2 indicates the DTD it references must be stored in the

```
<?xml version=“1.0” encoding=“UTF-8”?>
<!DOCTYPE catalog [
<!ENTITY NA “out of print”>
<!ELEMENT catalog (book)+>
<!ELEMENT book (author*, title*, date*, publisher*, city*)>
<!ATTLIST book
    binding CDATA #IMPLIED
    ISBN CDATA #REQUIRED>
<!ELEMENT author (#PCDATA)*>
<!ATTLIST author alive (yes | no) #IMPLIED>
<!ELEMENT title (#PCDATA)*>
<!ELEMENT date (#PCDATA)*>
<!ELEMENT publisher (#PCDATA)*>
<!ELEMENT city (#PCDATA*)>
]>
<catalog>
<book binding=“soft” ISBN=“0-1111-2222-3”>
<author alive=“no”>P.G. Wodehouse</author>
<title>The Mating Season</title>
<date>1949</date>
<publisher>H. Jenkins</publisher>
<city>London</city>
</book>
</catalog>
```

Figure 12.4 A DTD accompanying an instance document.

external file “**cat.dtd**.” Alternatively, as noted earlier, the contents of this DTD file could accompany the instance document itself. If that were the case, the document and the internal DTD could appear as shown in Figure 12.4. Here the DOCTYPE declaration line names the root element followed by a left bracket character instead of pointing to an external DTD file.

XML SCHEMA DEFINITION

The term *schema* has a broader context than just XML. With database systems, a schema is developed from the process of system analysis and the modeling of the database; the schema reflects the structure of a database and helps visualize or describe its organization. This is the general role of an XSD. There are excellent Web-based resources and tutorials available on creating XSDs (Vlist, 2001; W3Schools, 2008).

The DTD language was a holdover from XML’s roots in SGML. The W3C developed the XSDL in 2001 as an alternative to the DTD approach, and version 1.1 was released in August 2007 and the latest revision at W3C was in 2012 (Sperberg-McQueen & Thompson, 2008; W3C, 2012). As with a DTD, an XSD identifies the elements, child elements, nesting rules, cardinality, and attributes of a language but uses an XML itself to create the document model as opposed to the DTD language. The use of XML Schema has largely replaced the DTD approach to document specific languages based on XML. Some of the advantages and possible enhancements associated with the XML Schema approach are:

- Because the schema is XML based, a separate DTD language does not need to be learned. An understanding of XML is all that is needed to create them, and XML editors can be used for this purpose.
- They support additional data types that were not available in DTDs.
- Standard XML parsers can check for well-formedness and validity; no separate DTD reader is needed for intermediary processing.
- Common attributes across multiple elements can be defined in an attribute group.
- Namespaces can be used to share XML vocabularies and resolve name conflicts.

One of the primary functions of an XSD is to define names for elements and attributes, essentially creating a vocabulary. Before discussing the details of creating an XSD, it is helpful to introduce how *namespaces* and *namespace prefixes* support XML vocabularies because the examples given here all use the “xs” prefix. A namespace identifies a specific XML vocabulary, which is referenced in the XSD by the use of an assigned tag prefix. Prefixes are text strings followed by a colon that precedes the tag to form a qualified name. The default XSD tag’s prefix is **xs** or **xsd** as in the example **<xs:element name = “something” . . . >**. The use of the “xs” prefix ensures

that the word “element” is taken to be part of the XSDL vocabulary and does not have some other meaning associated with it. For instance, a chemistry markup language could use the word to refer to an “element” in sense of elements in the periodic table as opposed to elements in a XSD. Further, the namespace for the XSDL vocabulary is referenced as an attribute in the schema root element with an XML namespace reference, which has the form shown below. Namespaces will be discussed further later in this chapter.

xmlns:xs = “<http://www.w3.org/2001/XMLSchema>”

XML Schema Definition Example

All XSDs use the syntactical rules, but they can be expressed and organized in different ways. A simple schema can be developed by working backwards from the structure of an XML instance document. This is sometimes referred to as the “Russian Doll” method because elements and attributes are created in the XSD based on the nesting of elements observed within the instance document (W3Schools, 2008). This approach is relatively simple; however, other organizational views of the XSD are possible. The flat catalog view groups elements together by whether they are simple or complex. The class-based approach uses data types to organize the XSD. An instance document of the book catalog discussed earlier with DTDs is shown again in Figure 12.5, this time without the DOCTYPE SYSTEM reference to the “cat.dtd” file. This document sample will be used to demonstrate how to create an XSD using the “Russian Doll” approach discussed later in this section.

XSDL is itself an XML language, so the schema begins with the standard XML declaration, followed by the prefixed root element **schema** as shown in Figure 12.6.

This defines the root element as **schema**, and the tag prefix “**xs**” is assigned to indicate that all the elements in it are derived from the XSDL vocabulary.

```
<?xml version=“1.0” encoding=“UTF-8”?>
<catalog>
<book binding=“soft” ISBN=“0-1111-2222-3”>
<author alive=“no”>P.G. Wodehouse</author>
<title>The Mating Season</title>
<date>1949</date>
<publisher>H. Jenkins</publisher>
<city>London</city>
</book>
</catalog>
```

Figure 12.5 Book catalog XML instance document.

```
<?XML version="1.0" encoding="UTF-8"?>
<xs:schema>
...
</xs:schema>
```

Figure 12.6 The start of an XML Schema; root element is defined.

```
<xs:element name="some_element">
  <xs:complexType>
    <xs:sequence>
      ...
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Figure 12.7 Building the XML Schema—defining elements.

As noted earlier, the XSD accomplishes the same functions as a DTD; it must define the elements, their relationships, cardinality, and attributes. The XSDL syntax for defining elements and entities, as well as documenting their nesting rules, data types, attributes, and any numeric constraints on occurrences of elements, is different from the DTD element declaration format. Elements are defined in XSD in the form shown in Figure 12.7.

Elements are defined as either **complexType** or **simpleType**. Elements are complexType if they have child elements and/or attributes. SimpleType elements are those that contain only data; they can have neither child elements nor attributes. In the generic XSD shown in Figure 12.7, the element's name is provided as the value of the “**name**” attribute. Because it is the complexType, you can deduce it has either child elements and/or attributes defined. When child elements occur, they are nested within the **<xs:sequence>** tag pair. Each of the child elements themselves are either the complexType or simpleType. In addition, each element defined could have other attributes defined, for example, those that constrain its minimum or maximum occurrences or its data type. Once the XSD has been created, it is typically stored in an external file that has the extension “**XSD**.” In this example, the XML Schema created has been named “**cat-schema1.xsd**.” The contents of this file are in Figure 12.8; indentation is used to represent the nesting hierarchy of the elements visually.

Note that one complexType element in Figure 12.8 is handled differently from the others. The **<author>** element in this XSD has an attribute but does not have child elements. When a complex element has only attributes and no subelements, it is still a complexType, but the attribute is defined as “**simpleContent**.” This simple content information must be further refined by the **<xs:extension base = "some_datatype">** tag. The “**extension**” reference

```

<?xml version = "1.0" encoding= "UTF-8" ?>
<xs:schema xmlns:xs= "http://www.w3.org/2001/XMLSchema">
    <xs:element name= "catalog">
        <xs:complexType>
            <xs:sequence>
                <xs:element name= "book" maxOccurs= "unbounded">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name= "author">
                                <xs:complexType>
                                    <xs:simpleContent>
                                        <xs:extension base = "xs:string">
                                            <xs:attribute name = "alive" type =
                                            "xs:string"/>
                                        </xs:extension>
                                    </xs:simpleContent>
                                </xs:complexType>
                            </xs:element>
                            <xs:element name= "title" type= "xs:string"/>
                            <xs:element name= "date" type= "xs:date"/>
                            <xs:element name= "publisher" type="xs:string"/>
                            <xs:element name= "city" type= "xs:string"/>
                        </xs:sequence>
                        <xs:attribute name= "binding" type= "string" />
                        <xs:attribute name= "ISBN" type= "string" />
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
        </xs:complexType>
    </xs:element>
</xs:schema>

```

Figure 12.8 A “Russian Doll” style XML Schema (tested for validity at <http://jiggles.w3.org/cgi-bin/new-xmlschema-check>).

means the attribute that is declared is derived from, or *extends*, the data type given and indicates the type of information it carries, such as string text or an integer.

This XSD has been saved as “**cat-schema1.xsd**” in the same directory location as the instance document; that it is in the current directory location is deduced from the lack of any other path information to this file. The XSD is referenced in the document by including two pieces of information: first, a XSD exists that should be used to validate this document, and second, its location. These directives are passed to the parser with the information in attributes shown below:

```

xmlns:xsi = "http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation = "cat-schema1.xsd"

```

```

<?xml version = "1.0" encoding= "UTF-8" ?>
<!—two attributes have been added to the root element to indicate a schema is used and
where it is -->
<catalog xmlns:xsi= "http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation= "cat-schema1.xsd">
<book binding= "soft" ISBN= "0-1111-2222-3">
<author alive= "no">P.G Wodehouse</author>
<title>The Mating Season</title>
<date>1949</date>
<publisher>H. Jenkins</publisher>
<city>London</city>
</book>
</catalog>

```

Figure 12.9 XML namespace and Schema location reference added to the XML instance document.

These attributes are added to the root element, as shown in Figure 12.9. The “nonamespace” reference informs the parser that this document does not use a specific namespace but does use a schema for validation.

XML Schema Organization

The XSD format created from following a document prototype creates a valid model but results in a view of the XSD that is hard for people to read and follow. Alternate XSD presentations can provide a more organized view that is easier for people to understand and modify if needed. For a complicated XSD, one of the two organizational methods mentioned earlier is often used—the flat catalog approach that groups elements by simple or complex types and the class-based approach that is organized by data types. To illustrate how these methods can improve the layout and appearance of the XSD, the “**cat-schema1.xsd**” file has been reorganized using the flat catalog approach. This view depends on separating elements by type. This organizational view depends on a clear understanding of what makes an element either “simpleType” or “complexType.” As noted earlier, simple elements are those that contain only data; they cannot have child elements or attributes defined. complexType elements have child elements and/or attributes. Further, when a complex element has only attributes and no subelements, that attribute is defined as “simpleContent.” This designation uses the **<extension base = “some_type”>** tag to indicate the type of data it carries, such as text or an integer. The new view of the XSD has been saved as “**cat-schema2.xsd**” and is shown in Figure 12.10.

Note that this XSD accomplishes exactly the same functions as the first, but its structure is simplified by reducing the repeated nesting tag pairs that occur in the “Russian Doll” model, resulting in an XSD that is easier for people to read and logically follow. The simpleType elements are declared first, and those names are used to declare the complexTypes, which then use the **ref** attribute instead of **name**.

```

<?xml version = "1.0" ?>
<xs:schema xmlns:xs=
"http://www.w3.org/2001/XMLSchema">
<!-- first simple elements are grouped together -->
<xs:element name= "title" type= "xs:string"/>
<xs:element name= "date" type= "xs:date"/>
<xs:element name= "publisher" type="xs:string"/>
<xs:element name= "city" type= "xs:string"/>
<!-- now define the attributes -->
<xs:attribute name="alive" type="xs:string" />
<xs:attribute name= "binding" type= "xs:string" />
<xs:attribute name= "ISBN" type= "xs:string" />
<!-- finally, complex elements are defined -->
<xs:element name= "author">
<xs:complexType>
<xs:simpleContent>
<xs:extension base = "xs:string">
<xs:attribute ref="alive" type="xs:string" />
</xs:extension>
</xs:simpleContent>
</xs:complexType>
</xs:element>
<xs:element name="book">
<xs:complexType>
<xs:sequence>
<xs:element ref= "title" />
<xs:element ref= "date" />
<xs:element ref= "publisher" />
<xs:element ref= "city" />
</xs:sequence>
<xs:attribute ref="binding" type="xs:string" />
<xs:attribute ref="ISBN" type="xs:string" />
</xs:complexType>
</xs:element>
<xs:element name="catalog">
<xs:complexType>
<xs:sequence>
<xs:element ref= "book" maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>
</xs:element>
</xs:schema>

```

Figure 12.10 A catalog style organization of the XSD described earlier (tested for validity at <http://jiggles.w3.org/cgi-bin/new-xmlschema-check>).

VOCABULARIES AND NAMESPACES

The element and attribute names used in an XML implementation become the vocabulary of that specific language. These vocabularies are not predefined and are up to the creator of the language. However, XML takes a modular approach and encourages the use of existing XML vocabularies where they can be applied in a new context. Making use of existing vocabularies is obviously an efficient strategy, but it introduces the problem of possible name collisions or conflicts when identical names represent different things. For instance, the element name “title” used as a tag could refer to the title of a book, but it could also have different meanings in other vocabularies, perhaps referring to a person’s title in an organization, such as Dr., Director, Ms., or Professor. Similarly, an element that used the tag “table” could refer to a document object or a piece of furniture. If XML documents from different vocabularies are merged or if a single document depends on more than one vocabulary, the XML parser cannot resolve the different uses of the same tag without more information about its intended meaning. This is the function of namespace references.

Namespaces are referenced by binding an identifiable namespace to a particular vocabulary or to a specific prefix, and then adding that information to the document. The namespace identifier is in the form of a URI or a Uniform Resource Name (URN). A URI looks like the familiar URL; the URN reflects a name for a URI that has some institutional persistence. Either can serve as a unique identifier for the namespace, but each has its own syntax when used. Namespace declarations using a URI followed by one using a URN are shown here:

```
xmlns = “http://somelocation/XML/namespace/example”
xmlns = “urn:namespace_identifier:namespace_string”
```

Either of these statements, when added as an attribute to the root element, establishes a default namespace for that document. A prefixed namespace reference can also use either a URI or a URN and would take the form:

```
xmlns:bk = “http://somelocation/XML/namespace/example”
xmlns:bk = “urn:namespace_identifier:bk”
```

Both of these statements create a “bk” prefix that could then be used as an expanded name for any element in the document. For instance, the title tag would become <bk:title> to clarify that this instance of “title” is associated with the “bk” namespace. Certain letters are restricted and are not allowed in locally defined prefixes; the prefixes that start with the letter “x” are used to identify prefixes reserved by the XML standard.

Although namespace identifiers may look like URL addresses, the XML parser does not retrieve or reference the external namespace file. They are just a unique identifier for a namespace, and their “location” does not need to be an active or real URL Web address.

```
<catalog xmlns="http://somelocation/XML/namespace/example1">
<book binding="soft" ISBN="0-1111-2222-3">
<author alive="no">P.G. Wodehouse</author>
<bk:title xmlns:bk="http://somelocation/XML/namespace/example2">The Mating
Season</bk:title>
<date>1949</date>
<publisher>H. Jenkins</publisher>
<city>London</city>
</book>
</catalog>
```

Figure 12.11 XML namespace reference.

XML enforces rules about what namespace is in effect and where the namespace is applied; this is referred to as the *scope* of the namespace. Namespaces are inherited in a descending fashion through the tag hierarchy. A namespace declaration at any level affects all the child elements below the location of the declaration unless another namespace declaration at a lower level overrides it. The sample XML document in Figure 12.11 has two defined namespaces.

In the Figure 12.11 example, a default namespace of “**http://somelocation/XML/namespace/example1**” is established, and it applies to all the child elements that are nested within it. However, a second namespace is referenced that creates the prefix “**bk**,” and this associates the **title** tag with a different namespace. This prefix is available for use in any child element of the title element, but any element higher up in the hierarchy is outside the scope of this reference. Further, any child element of the title element that did not explicitly use the “**bk**” prefix is in the scope of the default namespace.

XML can be used to encode metadata and a namespace designation can identify the named location that contains the documentation or XML Schema associated with a metadata scheme. Dublin Core is one such metadata scheme, and the namespace statement for it would be:

xmlns:dc = “http://purl.org/dc/elements/1.1/”

This statement establishes both a namespace location as well as the prefix of “dc” for the Dublin Core element names such as “**dc:title**” and “**dc:creator**.” Steven Miller (2011) has an excellent how-to manual about the use of XML-encoded metadata for digital collections.

VIEWING AND PROCESSING XML

The universal client application for viewing Web content is, of course, the Web browser, and browsers were designed to handle HTML, not XML. Browsers vary in how they handle XML documents; generally, they simply display the full document instance and its accompanying markup in a nested hierarchical

display. Other technologies must be combined with XML to enable browser display beyond the default code only view.

CSS and XML

As discussed in Chapter 9, HTML benefits significantly from the application of CSS to achieve separation of content from presentation directives. XML has no stylistic elements, and it is completely dependent on style languages or other forms of programming to handle data presentation. You can apply CSS to the XML environment, but CSS is not a complete solution, and other programming must be utilized to achieve the true potential of XML.

Simple CSS rules applied to XML use XML tags as selectors instead of the HTML tags used in previous examples of CSS rules. Again using the book catalog example, you can add a processing instruction that references a style sheet called “**bk.css**” that is located in the “**styles**” subdirectory, as shown in Figure 12.12.

A browser view of the XML before and after the application of CSS is shown in Figure 12.13 followed by the appearance with the CSS applied in Figure 12.14.

XML file with a reference to CSS	Contents of “cat.css”
<pre><?xml version = "1.0" ?> <?xml-stylesheet type= "text/css" href= "styles/cat.css" ?> <catalog> <book binding= "soft" ISBN= "0-1111-2222-3"> <author alive= "no">P.G Wodehouse</author> <title>The Mating Season</title> <date>1949</date> <publisher>H. Jenkins</publisher> <city>London</city> </book> </catalog></pre>	<pre>catalog { background-color: #eeeeee; width: 100%; } book { display: block; margin-bottom: 30pt; margin-left: 0; } author { color: #ff0000; font-size: 18pt; } title { color: #000000; font-size: 18pt; } date { display: block; color: #0000ff; font-size: 18pt; } publisher { color: #ff0000; font-size: 18pt; } city { color: #000000; font-size: 18pt; }</pre>

Figure 12.12 Left: An XML document that references CSS. Right: The contents of the “cat.css” file.

A screenshot of a Microsoft Internet Explorer browser window. The address bar shows the path: C:\Users\Joe Miller\Documents\book-all.xml\cat-no.css. The main content area displays the following XML code:

```
<?xml version="1.0"?>
- <catalog>
  - <book ISBN="0-111-2222-3"
    binding="soft">
    <author alive="no">P.G
      Wodehouse</author>
    <title>The Mating
      Season</title>
    <date>1949</date>
    <publisher>H.
      Jenkins</publisher>
    <city>London</city>
  </book>
</catalog>
```

Figure 12.13 A browser view of an XML document without CSS applied.

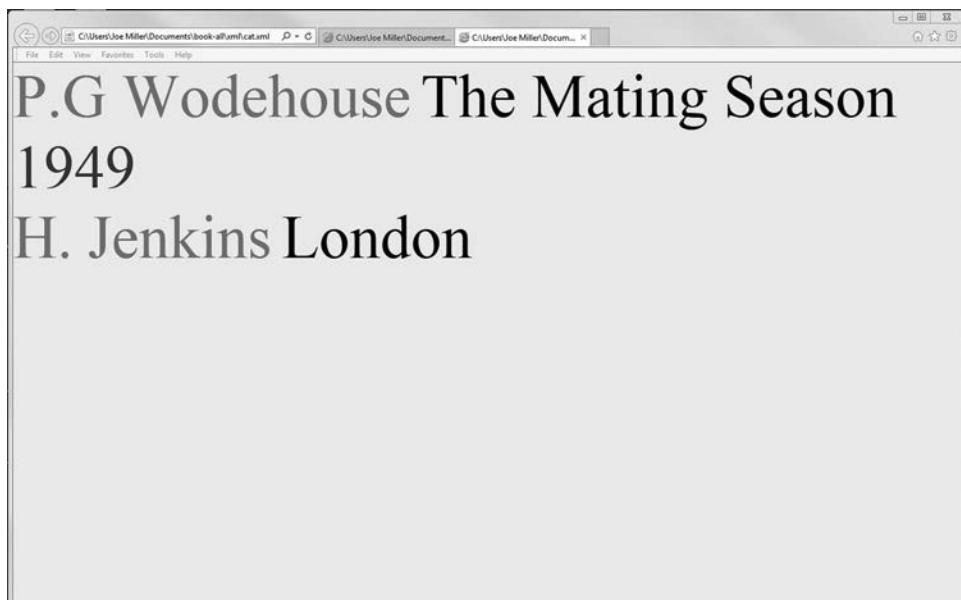


Figure 12.14 A browser view of an XML document with CSS applied.

XSL

CSS is relatively easy to learn, and most browsers can utilize it, but it is not a true programming language. It therefore has some limitations in the context of XML, specifically: (1) XML elements must be displayed in the order in which they occur and cannot be rearranged as needed. (2) CSS cannot handle computations or logical “if-then” conditional tests, and it cannot dynamically generate text blocks. (3) CSS is limited to utilizing simple parent-to-child relationships in its organization. XSL has been developed by the W3C as a family of related technologies to address document transformation and presentation in a more robust way (W3C, 2013). The XSL was introduced on October 2001, and version 1.1 was released as a W3C recommendation in December 2006. According to W3C, XSL is a set of three inter-related technologies:

- XSL Transformations (XSLT), a language for transforming XML;
- XML Path Language (XPath), an expression language used by XSLT to access or refer to parts of an XML document; and
- XSL Formatting Objects (XSL-FO), an XML vocabulary for specifying formatting semantics. (W3C, 2008a, para. 1–2)

XSL is a programming language with support for built-in functions and features that make it more sophisticated than simple CSS. The two main components are XSL-FO, which allows for text properties and formatting similar to CSS, and XSLT, which allows for the transformation of XML documents to other formats such as HTML or PDF.

XML-RELATED TECHNOLOGIES

In addition to XSL, there are other technologies a more comprehensive treatment of XML could consider. These related topics are not discussed in detail beyond the brief descriptions listed here, but those interested in creating actual XML applications would want to explore them further. A partial list of related technologies includes:

- The DOM has been discussed in several areas of this text. Generally, DOM refers to a hierarchically structured object-centered view of a document. This view of DOM is codified within the 1998 Level 1 DOM Specification at W3C, described as “a platform-and language-neutral interface that will allow programs and scripts to dynamically access and update the content, structure and style of documents” (W3C, 1998). New features have been added to this model with the release of Level 2 in 1999, which added CSS and query support. Level 3, released in 2004, allowed for programs and scripts to dynamically access and modify content. In the context of XML applications, DOM is a collection of routines, procedures, and tools that exploit

the tree structure inherent in XML to build XML-processing software (Harold & Means, 2002). Such a collection of routines and tools is referred to as an API or Application Program Interface.

- The Simple API for XML (SAX) is an event-based API originally designed as Java API, which is probably why so many books on XML also attempt to cover the Java programming language. Many XML parsers and validators are developed with the Java programming language.
- The SOAP is defined at W3C as “a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralized, distributed environment using XML” (W3C, 2000). SOAP creates messages and exchanges them between applications in a stateless fashion, usually via HTTP. Essentially, it is one of several Web service models that facilitate interoperable machine-to-machine program interactions. This protocol consists of three components: (1) the SOAP envelope that defines the message, (2) the encoding rules for defining data types, and (3) the SOAP RPC representation. RPC stands for the Remote Procedure Call that controls one application or can execute another on a remote server.
- SVG version 1.1 is a 2003 W3C language specification for describing two-dimensional graphics and graphical applications in XML (W3C, 2008b).
- Asynchronous JavaScript and XML (AJAX), as the name implies, uses JavaScript and the AJAX engine to send data asynchronously between client and server, facilitating interaction with a page and resulting in reloading specific components without having to retrieve the full page. AJAX has application to both dynamic HTML as well as XML applications.
- RDF is an XML-based metadata interoperability standard and is viewed as one of the key components of the semantic Web. Given its importance in semantic search discussed in Chapter 15, it is explored further in the section that follows.
- XHTML is a reformulation of HTML to comply with the stricter requirements of XML. XHTML is also given more attention in the next section.

XML IMPLEMENTATIONS

There are a large number of XML-based languages and standards in use, and as noted earlier, it is considered a key element in the development of the semantic Web. XML is the foundation of the various RSS news feeds and blogs. RSS has different formats, and the acronym has several definitions, including Really Simple Syndication, Rich Site Summary, or RDF Site Summary, depending on the source. Other XML implementations include specialized languages like MathML for handling equations and the Encoded Archival Description (EAD) language used to create finding aids in library special collections. Library standards based on XML also include the Metadata Object Description Schema (MODS) and the Metadata Encoding and Transmission

Standard (METS) at the Library of Congress. These XML implementations are of particular importance to those building and maintaining digital libraries, repositories, and archives.

RDF and RDFa

RDF (Resource Description Framework) is an XML application that associates a resource description and the resource through a URI. The RDF description is based on identifying the set of essential resource properties, each in the form of a property type and value pair. RDF allows machine understanding of human statements—RDF can represent any semantically meaningful statement comprised of a subject-predicate-object as a *RDF triple*. For example, each part of a RDF triple is shown in the statement “<That photo> <is licensed under> <a Creative Commons Attribution license>” (RDFa, 2013). There are parallels between RDF concepts and those of relational databases and the ERD model that was discussed in Chapter 11. The Entity-Relationship-Diagram was described as a way to model a database by identifying the entities, their attributes, and entity relationships, and a primary key is used to identify a record. RDF also represents information about entities, attributes, and relationships. However, the RDF data model is very different from the RDB table-based data model; the RDF data model is that of a directed graph, where the nodes represent either a subject or an object and the connections, referred to as edges, represent the predicate of a triple (Resource description framework, 2004).

RDF is a language and an abstract data model for representing metadata. It can be expressed in XML (RDF/XML) or with the more compact Turtle (Terse RDF Triple Language) format. RDF/XML has been used to create specific RDF Schemas (RDFS). With RDFa, it is also possible to embed metadata within existing HTML or XHTML in the form of new element attributes. The RDF standard has a default namespace that can be extended to include one or more specific metadata namespaces. For instance, the Dublin Core (DC) metadata scheme was developed by OCLC to facilitate simplified cataloging of Web resources. DC contains the type of metadata elements typically found in a library catalog such as author, title, description, and so forth. An RDF container could include a DC reference as an identified prefix as shown here:

<rdf:RDF xmlns:rdf= “<http://www.w3.org/1999/02/22-rdf-syntax-ns#>” xmlns:DC= “<http://purl.org/DC#>”>. The DC prefix would then qualify elements as being in the DC set for cataloging, such as **<DC:author>** or **<DC:title>** (W3C, 2004).

As will be discussed further in Chapter 15, RDF is a key component of the semantic Web and semantic search (Amerland, 2014). RDFa (Resource Description Framework in Attributes) also supports semantic search by allowing metadata to be directly embedded within HTML5, XHTML, and XML content, providing machine-readable metadata in the form of property-value pairs (RDFa primer, 2013). RDFa accomplishes this with new attributes that can be added to specific HTML elements or used with a generic content container such as the DIV or SPAN tags. RDFa has defined attributes for each part of a RDF triple: for example, the attribute **about** is for a subject, the attributes of **property**, **typeof**, **rel**, and **rev** are for predicates, and attributes **content**,

resource, and **datatype** can represent objects. A few examples of RDFa attributes and their uses are below:

- The **property** attribute, used to provide additional information about the link or content. Example: `` provides information that this is a specific kind of link, not just the location of the href. Links with this added information are referred to as a “flavored links.”
- The **vocab** attribute, used to establish a default vocabulary. Example: `<body vocab = "http://purl.org/dc/terms/">` establishes the DC vocabulary as the default namespace for any other property attributes used through the document.
- The **content** attribute, used to provide the same information in the tag as is being marked up in the document, but in a required format such as a date format. Example: `10th of September, 2013`
- The **rel** and **rev** attributes are used to establish a relationship (or a reverse relationship). Example: `cc`

Discussion of how these technologies relate to search is explored in Chapter 15.

XHTML

The XML standard has influenced the development of HTML. Because many predict the Web may make significant use of XML in the future, it makes sense to prepare current HTML documents to facilitate that transition. The XHTML standard is a reformulation of HTML to make it comply with the more rigorous rules of XML. Essentially this eliminates much of the “bad” HTML that has been acceptable on the Web and is permitted in most browsers. XHTML will ensure that new Web pages are both well formed and valid. The main rules that XHTML enforces that HTML is lax about are:

- Documents must begin with a DOCTYPE declaration that references one of the three XHTML DTDs, strict, transitional, or frameset. The strict DTD disallows deprecated tags and attributes allowed by the looser transitional DTD. The frameset DTD is the same as the transitional with added frameset and iframe tag support.
- The root HTML element must be present.
- Tags and attributes must be written in lowercase.
- All tags, including empty elements, must have an end tag.

- All attribute values must appear within quotation marks.
- Nesting order is strictly enforced; any tag nested within another must also end within the parent tag.

In anticipation of more XML content and applications, all Web content produced with HTML should ideally be compliant with XHTML standards. For preexisting content created in the more forgiving world of simple HTML, there are programs that can “tidy up” the markup and attempt to auto-convert it to comply with the XHTML standard.

ALTERNATIVES TO XML

XML can support data structuring and data exchange between programs and it supports many Web 2.0 technologies, such as RSS feeds and AJAX, described earlier in this chapter. However, it has some disadvantages; it tends to be overly verbose (i.e., requires a great deal of markup overhead), it requires the creation of a DTD or XML Schema for validity, and it needs specialized programs for parsing the markup. Consequently, some Web developers and programmers view it as overkill for many situations. Alternatives to full XML implementations for data interchange include JSON, the JavaScript Object Notation format, discussed further below, and YAML. YAML is one of those recursive acronyms for “YAML Ain’t Markup Language”) and it is another human-readable format that combines concepts from XML and various other programming languages such as Python and C++.

JSON

JSON format (www.json.org) was developed by Douglas Crockford as a way to manage Web application data interchange. It was derived from JavaScript (hence the name) but it is independent of JavaScript; the JSON format is supported in many languages including Python, PHP, Java, C++, and Ruby (Sripurasra, 2013). It is a lightweight, human-readable text format that does not need a separate XML parsing program. JSON data can be handled directly by outputting data from JavaScript variables, such as with the document.write function. JSON text can be retrieved from a server and converted into objects with the eval() function built into JavaScript interpreter or alternatively, with a separate JSON.parse function to create script objects (JSON Tutorial, 2013). However, the eval() function can introduce security risks such as malicious JavaScript insertion because it compiles any code in the text; the JSON.parse, which only processes JSON data, or a JQuery option are considered more secure. Using HTTP POST instead of GET for JSON data requests is also recommended (Data security with JSON, 2013). Many services such as Reddit, Twitter, Facebook, and Flickr use JSON-based APIs for data interchange (Tamim, 2013). JSON is also a good fit for many AJAX applications, where it can be more efficient than XML for data interchange; XML data must be extracted with the XML DOM and then stored, whereas JSON data can be fetched and utilized directly in a script on the client or on the server (JSON Tutorial, 2013).

Those interested in AJAX development with JSON-based feeds will encounter a variation called JSON-P (for “JSON with padding”). This version of JSON was designed to address a problem that occurs with AJAX calls across different domains. To prevent malicious script behavior, browsers generally enforce a “same-origin policy” that allows data exchange only when the target resource domain is the same as that of the page originating the request. JSON-P uses the HTML script tag to handle this cross domain issue. The details of JSON-P implementation is beyond the scope of this overview, but Sriparasa (2013) and JSON-P.org have more on this topic.

XML data can be converted into JSON format manually, but automatic conversion is also possible with XSL transformations or JavaScript (Azad, 2007). As with XML, there is a JSON Schema that is similar to XML Schema, and as with XML schema files JSON text can be checked for errors using one of many online validators. The syntax of JSON is similar to that of JavaScript and uses the same punctuation (see Chapter 10); it uses curly brackets {} to identify objects, and square brackets [] to identify arrays, which are used to accommodate the nesting structure of XML. Within each object or array, XML elements and attributes appear as sets of key/value pairs. Each key property name and value is enclosed in double quotation marks and each property and value is separated by a colon; multiple pairs are separated within an object or array by commas. For example, the XML document example in Figure 12.5 can be converted to the JSON format. That XML instance document is compared to its JSON version in Figure 12.15.

Although JSON data format is usable by many scripting languages, JavaScript is a common choice. JavaScript is relatively easy to learn, but there are also readymade JQuery libraries available that support JSON Web features, such as creating a Flickr photo stream (Lengstorf, 2009). A simple example of outputting data from the JSON data set shown in Figure 12.15

<pre><?xml version="1.0" encoding="UTF-8"?> <catalog> <book binding="soft" ISBN="0-1111-2222-3"> <author alive="no">P.G. Wodehouse</author> <title>The Mating Season</title> <date>1949</date> <publisher>H. Jenkins</publisher> <city>London</city> </book> </catalog></pre>	<pre>{ "catalog": [{ "title": "The Mating Season", "ISBN": "0-1111-2222-3", "binding": "soft", "author": { "name": "P. G. Wodehouse", "alive": "no" }, "publisher": "H.Jenkins", "date": "1949", "city": "London" }] }</pre>
---	--

Figure 12.15 The book instance XML document on the left compared to its JSON equivalent on the right.

<pre><script> var books = JSON.parse('{"catalog": [{"title":"The Mating Season", "ISBN":"0-1111-2222-3", "binding":"soft", "author": {"name": "P. G. Wodehouse", "alive": "no"}, "publisher": "H.Jenkins", "date": "1949", "city": "London"}]}'); alert(books.catalog[0].author.name); document.write(books.catalog[0].title); </script></pre>	<p>Converts JSON strings into JavaScript objects stored under name “books.”</p> <p>Catalog object is an array within square brackets with property/value pairs. Author object has two properties defined.</p> <p>Displays alert window with value “P.G. Wodehouse”</p> <p>Prints title “The Mating Season” into page.</p>
--	---

Figure 12.16 Example of JSON data in a JavaScript in a Web page script on the left with a description of its action in a browser on the right.

is given in Figure 12.16. The choice between using XML and an alternative such as JSON depends on both the needs and preferences of the Web developer.

SUMMARY

XML is a meta-language that has many immediate applications and is likely to become increasingly important in the future Web. The core idea of XML is to facilitate the creation of languages to structure content in ways that are not possible in simple HTML and to facilitate data exchange among programs. The two main approaches to defining an XML implementation are through the creation of a DTD or an XSD. Both techniques formally define the elements, attributes, the nesting rules, and their numeric constraints. However, DTDs are an older technology carried over from the SGML standard, and XSDL represents a newer and generally preferred approach to creating a formal document model with a XSD written according to the rules of XSDL. An XSD can be created by “working backwards” from a sample document using the “Russian Doll” strategy, but there are other ways to organize and present an XSD, such as with a flat catalog view or with a class-based approach that uses data types as the organizational theme. Because XML is extensible, the vocabularies for different languages are not predefined. Words used to identify elements may overlap with other vocabularies where the term has a different intended meaning, so namespaces provide a mechanism to identify and separate these vocabularies.

The presentation and display of XML documents are problematic in Web browsers because these languages are not HTML. A browser can only show the XML code, but CSS can be used to provide formatting of the XML document when viewed in a browser. However, CSS has limitations, which are addressed in the more powerful programming language of the XSL specification.

There are many XML-related technologies and standards, such as XHTML, DOM, MODS, RDF, SAX, and AJAX. As highlighted in Chapter 11, XML-encoded metadata is essential to the OAIS model and many digital library and repository initiatives. RDF and RDFa come up again in Chapter 15 in the context of semantic search. Web developers have alternatives to XML for Web application data exchange, such as JSON and YAML that may be more efficient for some applications such as AJAX. With this brief overview of the world of XML, it is apparent that a discussion of this environment quickly leads to many related topics for further exploration.

REFERENCES

- Amerland, D. (2014). *Google semantic search: Search engine optimization (SEO) techniques that gets your company more traffic, increases brand impact and amplifies your online presence* (1st ed.). Indianapolis, IN: Que.
- Azad, K. (2007, January 11). Using JSON to exchange data. Retrieved December 9, 2013, from <http://betterexplained.com/articles/using-json-to-exchange-data/>
- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.
- Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic Web. *Scientific American*, 284(5), 10.
- Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F., & Cowan, J. (2006). Extensible markup language (XML) 1.1 (2nd ed.). Retrieved May 1, 2008, from <http://www.w3.org/TR/xml11/#sec-xml11>.
- Data security with JSON. (2013, January 23). Retrieved December 12, 2013, from <https://www.golemtechnologies.com/articles/data-security-and-json>.
- Graves, M. (2002). *Designing XML databases*. Upper Saddle River, NJ: Prentice Hall.
- Harold, E. R., & Means, W. S. (2002). *XML in a nutshell* (3rd ed.). Beijing: O'Reilly.
- JSON Tutorial. (2013). Retrieved December 9, 2013, from <http://www.w3schools.com/json/default.asp>.
- Lengstorff, J. (2009). JSON: What it is, how it works, & how to use it. Retrieved December 9, 2013, from <http://www.coptelabs.com/blog/json-what-it-is-how-it-works-how-to-use-it/>
- Miller, S. J. (2011). *Metadata for digital collections: A how-to-do-it manual*. New York: Neal-Schuman Publishers.
- Resource description framework (RDF) model and syntax specification. (1999, January 5). Retrieved January 4, 2014, from <http://www.w3.org/TR/PR-rdf-syntax/>
- RDFa. (2013, October 28). Retrieved January 4, 2014, from <http://wiki.creativecommons.org/RDFa>.
- RDFa 1.1 primer—Second edition: Rich structured data markup for Web documents. (2013, August 22). Retrieved January 4, 2014, from <http://www.w3.org/TR/xhtml-rdfa-primer/>
- Sperberg-McQueen, C. M., & Thompson, H. (2008, July). W3C XML Schema. Retrieved September 27, 2008, from <http://www.w3.org/XML/Schema>.
- Sriparasa, S. S. (2013). *JavaScript and JSON essentials paperback*. Birmingham, UK: Packt Publishing.
- Tamim, Z. (2013, August 31). How to Fetch and Parse JSON Using iOS SDK. Retrieved December 12, 2013, from <http://www.appcoda.com/fetch-parse-json-ios-programming-tutorial/>

304 Internet Technologies and Information Services

- Vlist, Eric van der. (2001, October 17). Using W3C XML Schema. Retrieved August 30, 2013, from <http://www.xml.com/pub/a/2000/11/29/schema/part1.html>.
- W3C. (1998, October 1). Document object model (DOM) level 1 specification. Retrieved August 30, 2013, from <http://www.w3.org/TR/REC-DOM-Level-1>.
- W3C. (2000, May 8). Simple object access protocol (SOAP) 1.1. Retrieved August 30, 2013, from <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- W3C. (2001). Semantic Web activity. Retrieved October 1, 2007, from <http://www.w3.org/2001/sw>.
- W3C. (2004, February 10). RDF primer. Retrieved December 15, 2013, from <http://www.w3.org/TR/REC-rdf-syntax>.
- W3C. (2010). Scalable vector graphics: XML graphics for the Web. Retrieved December 15, 2013, from <http://www.w3.org/Graphics/SVG>.
- W3C. (2012, April 5). W3C XML schema definition language (XSD) 1.1. Retrieved May 13, 2014, from <http://www.w3.org/TR/xmlschema11-1/>
- W3C. (2013, February 8). The extensible stylesheet language family (XSL). Retrieved December 15, 2013, from <http://www.w3.org/Style/XSL>.
- W3C. (2013, February 7). Extensible Markup Language (XML) 1.0 (Fifth Edition). Retrieved May 13, 2014, from <http://www.w3.org/TR/REC-xml>
- W3C. (2013, June 27). SemanticWeb. Retrieved May 15, 2014, from <http://www.w3.org/wiki/SemanticWeb>.
- W3C. (2013, December 11). Semantic Web activity. Retrieved December 31, 2013, from <http://www.w3.org/2001/sw>.
- W3Schools. (2013). Introduction to XML schema. Retrieved December 15, 2013, from http://www.w3schools.com/schema/schema_intro.asp.
- Yott, P. (2005). Introduction to XML. *Cataloging & Quarterly*, 40(3/4), 213–235.

ADDITIONAL READING

- Fitzgerald, M. (2004). *XML hacks: 100 Industrial-strength tips & tools*. Beijing: O'Reilly.
- Ray, E. T. (2003). *Learning XML*. Sebastopol, CA: O'Reilly.
- Rockwell, W. (2001). *XML, XSLT, Java, and JSP: A case study in developing a Web application*. Indianapolis, IN: New Riders.
- XML Guild. (2007). *Advanced XML applications*. Boston, MA: Thomson.

WEBSITES OF INTEREST

- EAD at <http://www.loc.gov/ead>.
- RDF at <http://www.w3.org/TR/REC-rdf-syntax>.
- RDFa at <http://www.w3.org/TR/xhtml-rdfa-primer>/
- Tools for working with RDF at <http://www.linkeddatatools.com/index.php>.
- SOAP at <http://www.w3.org/TR/2000/NOTE-SOAP-20000508>.
- XML at <http://www.w3.org/XML>.
- XML Schema at <http://www.w3.org/XMLSchema>.



Part

3

Internet Content and Information Retrieval

Part 3 is devoted to the topic of Internet information resources and their retrieval using Web search services. The section begins with an examination of the sources and types of content available on the Internet and the various file formats in which it resides. Chapter 14 is an overview of textual information retrieval concepts and models, which provides the foundation for Chapter 15 on Internet search engines. That chapter includes a brief history of Internet searching; an examination of search engine functions and limitations; a description of the process they use to create a searchable index; relevance ranking issues; link analysis applications; personalization and visualization techniques; and semantic search. This section concludes with a chapter examining the impact of the Internet on libraries and the information professions, exploring new applications and opportunities presented by Web 2.0 technologies and beyond.

This page intentionally left blank

Internet Content

Internet technologies are exciting, but *content* is what has driven much of the success of the Internet. There is a plethora of file types and formats making up the diverse forms of content accessible via the Internet. Many content forms are static files; other content sources are dynamic pages drawn from a database or other sources. Content residing in static forms is characteristic of Web 1.0; dynamic content production is more characteristic of Web 2.0 approaches. However, even with the rise of dynamic sources, a huge volume of Web content is still stored on servers in the form of autonomous files that can be requested by Web client software. These static formats are important because they comprise much of the indexable Web that is accessible via search engines. These formats include a huge number of HTML files as well as a variety of other documentary and multimedia file types processed directly by Web browsers or by a helper/plug-in application. Collaborative content management approaches have become increasingly popular, and software for wikis, blogs, and podcasts makes it easy for groups of content specialists to create and modify Web pages without having to master all the underlying technical issues of HTML markup and server access. This discussion of static, indexable Web content begins with a technical description of file basics and then examines the major important file types for Internet-delivered documentary and multimedia content.

Internet-based content has become a primary information resource for many, which has raised concerns about the quality of these sources. Many resources found as the result of Internet search are of questionable quality and authority, and librarians and educators have responded by seeking to promote information literacy skills. Information literacy is a necessary and critical skill set for information seekers in the twenty-first century. Information must not only be found, but it must also be evaluated and judged for its appropriateness to an information need. This chapter on Internet content types therefore concludes with discussion of some general guidelines for evaluation of Internet content.

CONTENT CHALLENGES

Preparing content for presentation on the Web involves both technical and epistemological challenges. The technical issues include preparation of content for migration to the Web, deciding whether the content will reside in static files or be generated dynamically and identifying the best format for content delivery to a mass audience. In addition to format decisions, epistemological issues exist that require determining what the resource is about, how to represent it, and its ultimate usefulness to the user.

The focus of this discussion is primarily on static content formats, but a broader discussion of Internet content could explore a number of related issues that present legal, policy, and ethical dilemmas for information consumers and libraries. The vast amount of “invisible Web” content, the challenges and controversies surrounding Internet content considered inappropriate (e.g., pornography and hate speech), and the possibility of deliberate misinformation make finding and using Internet information problematic. Another content challenge is that the global Internet is multilingual, and many excellent resources may be unavailable in the preferred language of the user. In addition, recent restrictions that limit or prevent access to previously available government information raise freedom of information issues. A few of these issues are explored in this chapter in the section on content evaluation. However, the focus of this chapter is on technical aspects of content formats, not on these related ethical and policy issues.

FILE BASICS

Much Internet content resides within static file containers that reside on Web servers and are available for delivery on request through client–server HTTP interactions. A file is a container of digital information stored and retrieved under some name. Most, but not all, computer systems view the digital information that makes up a file as a stream of bytes each containing 8 bits. Files have two key properties: they are *portable* and *persistent*; that is, they can easily be copied or moved from one location to another, and they do not disappear when the computer is turned off.

Although most file types are viewed as a stream of bytes, how a program interprets those bytes can vary. By convention, computer files are either *text files* or *binary files*. In ASCII text files, each byte value is associated with a character in an established code. ASCII is a 7-bit code where the eighth bit is used for error checking; extended ASCII uses all 8 bits in the code set. An 8-bit code permits a character set of 256 symbols; for non-Western character sets, different versions of the Unicode standard have 16 or 32 bits for character addressing, representing much larger sets of possible symbols.

Binary files (8-bit files) contain byte values that are not intended to be associated with ASCII characters; these bytes perform other functions that provide instructions or control codes to a program. Programs that create binary file types require the same program for successful viewing. For instance, Word documents, Excel spreadsheets, or Access databases are all proprietary Microsoft formats that require those programs for subsequent access to the content.

A simple test for text files is to attempt to view the contents of any file in a text editor such as Notepad; a text file is readable as text, but a binary file produces random odd-looking characters. The large numbers of proprietary formats represent the various requirements developers must consider when creating a new program. These include file size constraints, the speed of writing and reading the file to disk, and the possible need for random access to different parts of the file. Most Windows PC files are identified to programs and users alike by a naming convention that assigns a standard three or four-character extension to the filename. An OS then associates extensions with the appropriate application.

Proprietary formats are well adapted to the particular needs of a program and result in highly efficient, compact containers that preserve the content along with data structuring and presentation information. The downside of proprietary formats is that they present barriers to those who do not have the same program that created them, or some converter or reader program. For instance, the world of word processing now dominated by Microsoft Word was once much less homogenous; people in a work group might have used WordStar, WordPerfect, Word, or some other preferred program. This made data exchange problematic and often resulted in the need to exchange documents in the lowest common denominator format of simple ASCII text. Files that needed to be edited and shared by many types of computers often were saved in text formats such as TXT (text), RTF (Rich Text Format), or XML.

The Internet is designed to facilitate information sharing using text protocols, making the ASCII standard a natural choice for Web documents. All the Internet protocols, including the Web, move information as streams of 8-bit characters that are interpreted according to the ASCII standard, and encoding schemes are required to facilitate transfer of proprietary binary formats. The majority of Web content is in the HTML format; these files are a subset of the universe of text files. However, the choice of a simple text file format for Web pages presents its own set of tradeoffs. On the positive side, text files are very portable and do not require expensive proprietary software; in addition, all operating environments have some form of text editing capability such as the vi or emacs editor on UNIX systems, the Edit program with DOS, and Notepad in Windows. The downside of text formats is that presentation information and logical structure available in other formats can be lost. HTML and XML address this issue by using tags that are themselves simple text but inform the displaying program how to present the data, define document structural elements such as titles or headings, and in the case of XML, provide data structuring.

STANDARDS AND FORMATS

In an open environment like the Internet, the priority given to portability led to a handful of formats becoming the standards for documents and accompanying graphics. How do such standards come about? Standards might really just reflect accepted practice, which does not necessarily imply a best practice has been adopted. Ideally, standards are the result of expert opinion and are codified through adoption by some standards organization, but they may

also be de facto standards that emerge from common use. For instance, the GIF format commonly found on the Web actually began as a proprietary format, the CompuServe graphics interchange format. This format made use of a patented compression technology owned by Unisys, which resulted in licensing issues that were eventually resolved. The GIF format was adopted early on in Web publishing, and the capability to display GIF was incorporated into all graphical browsers. A newer format, the PNG, came about through the efforts of a committee seeking to create a better open standard for Web graphics that would be free of the initial licensing constraints associated with GIF; Chapter 7 has more discussion on these graphics formats. HTML is an open standard, but another common Web format, PDF, was developed and owned by Adobe. As of July 2008 it became an ISO (International Organization for Standardization) open standard. PDF requires the free Adobe reader software for display and PDF files can be created with Adobe Acrobat software or with other options such as PDF Creator from SourceForge.net (<http://sourceforge.net/projects/pdfcreator/>).

PRESENTING CONTENT ON THE WEB

When planning to present content on the Web, a publisher must consider a number of questions about the best way to represent the content digitally and the most appropriate final output format for its intended audience. The first issue depends in part, on whether the content currently exists in a digital version; perhaps it is in a simple text file or in some proprietary file format. If there are digital versions, the next step is to determine if the program that created the file format is still available; in the case of older formats, the lack of an appropriate program to display it could render the file useless. If there are no digital versions and the content is available in hard copy only, there are other issues to consider, including the quality of the original as well as whether a digital image of the original is sufficient or if an editable text version is required. If editable text is needed, one option is to use OCR (optical character recognition) software to create a digital copy, but this option depends on the quality of the copy to be scanned. If a document image is created, decisions must be made regarding the appropriate scan resolution needed to give an adequate rendering of the document. The most popular solution to emerge is scanning to PDF, which can retain both image appearance and searchable text capabilities.

Once some of the input issues are resolved, the next step is to consider the output format. If the digitized document is presented as an image, the format choices are limited to those that can be displayed in a Web browser (GIF, JPG, PNG, and perhaps SVG). However, presenting textual content in an image format requires high-resolution scanning to make the text readable, resulting in very large file sizes. Image views of text are more commonly used for older, handwritten documents where the goal is to reproduce the look of the original. Proprietary document formats such as MS Word may be an option for some content if the target audience can be expected to have that program. However, because proprietary formats are problematic for many users, a more appropriate output format would be HTML, viewable in any browser, or a scan to PDF,

which only requires the installation of the free Acrobat Reader software. Each choice has its own set of costs and tradeoffs. The final choice of a format is up to the content producer who must consider the needs of the target audience, the nature of the resource, and whether the resource is intended for display on a screen or destined for some other output device such as a printer. For instance, many forms that were once filled in manually are now made into HTML forms or PDF files that allow the form to be completed online and then printed out or saved. Format choices are also influenced by expectations of how the document is to be used; for instance, the Web is often used as a means to provide access to documents that are meant for printed output as opposed to screen viewing.

COMMON DOCUMENT FORMATS

The HTML format is by far the most common file type on the Web. Nevertheless, there are many other specialized formats encountered on the Web. The Adobe PDF is common and has become a de facto document standard. PDF is similar to the document description language used in Adobe PostScript but is better suited for online display and browsing. PostScript is a “page description language” with instructions to a PostScript device on how to format the content. PostScript documents are intended for output on a PostScript printer that has the program to interpret these instructions. Many word processors and desktop publishing programs can convert internal fonts to PostScript instructions, which results in a consistent layout in the finished product that is device independent. The PostScript format is great for document description but not very good for document interchange. When this format is encountered on the Internet, the intent is usually to make a document available for download and subsequent printing on a PostScript printer. The common extensions for these files are **PS**, **EPS**, **PFA**, and **PFB**. When viewed in a text editor, the first line of these files is a comment line that identifies the specific file type.

PDF has a hierarchical structure and a document directory that allows random access to different parts of the document. PDF files use the **PDF** extension and are technically a type of text-based format but are transferred as binary files. As with postscript files, the first line in the file contains the PDF version in a comment line if it is opened in a text editor. While the reader program is a free download, PDF files require the Acrobat software or other software such as PDF Creator to create or modify PDF files or convert Word documents to PDF. Now that PDF is an open standard published by ISO, there are fewer ownership concerns with its use.

eBooks

eBooks are a special type of document available on the Web through commercial outlets, libraries, and in collections of works in the public domain. The popularity of eBooks has grown in tandem with the increasing use of mobile devices and inexpensive dedicated readers. PDF is a common format for

312 Internet Technologies and Information Services

eBooks but there are specialized formats as well, many of which have been derived from Web technologies such as HTML, XHTML, CSS, and JavaScript.

TABLE 13.1 Common Document Formats Found on the Web

Extension	Format	Description
HTML	Hypertext Markup Language	Text files marked up in HTML to become static Web pages
SHTML	HTML page with some dynamic content; usually implies some SSI	A SSI is usually some value for a defined variable, such as an environment variable (like date/time).
ASP	Active Server Page	ASPs are Web pages that contain programming in Visual Basic or JavaScript. When a server gets a request for an ASP, it executes the embedded code. Used as an alternative for CGI scripts, which allow interactivity with databases, etc.
PDF	Adobe portable document format	Similar to postscript but designed for portability and exchange
EPUB	Electronic Publication	The standard developed by the International Digital Publishing Forum (IDPF); based on open technologies such as XHTML, CSS, and JavaScript. DRM is available
MOBI	Mobipocket eBook format	Based on HTML, XHTML, and CSS
AZW	Amazon format	Proprietary Amazon format based on MOBI
KF8	Kindle Format	Release coincided with the Kindle Fire and supports HTML5 and CSS3 features
TXT	Text	Simple ASCII text file
RTF	Rich text format	Text format with some formatting retained
CFM	Cold fusion	Adobe (formerly Macromedia) format for Web programming applications
DWF	Design Web Format	Autocad drawings

In addition to the structural differences among file types, eBooks often also employ DRM (Digital Rights Management) technologies that control who can access the file and for how long. Just as with all other computer file types, differences among eBook formats determine the specific technologies needed for access and display. Common eBook formats include the various **EPUB** types (EPUB2 and EPUB3), **Mobipocket** (MOBI), Amazon's **AZW** and **Kindle Format 8** (KF8), Apple's **iBooks Author**, and Barnes and Noble's **NOOK Kids**. The EPUB formats are based on the Open eBook and XHTML standards. MOBI, developed by a French company, was an early format derived from HTML and CSS; it was purchased by Amazon in 2005 but Amazon has since replaced it with the newer AZW and KF8 file types. The NOOK Kids format is similar to ePUB but it is based on PDF rather than on HTML and CSS. The proprietary iBook format from Apple is similar to ePUB but it is also different in its implementation, so these formats are not interchangeable (eBook formats, 2014). A few of these other document formats are listed in Table 13.1.

COLLABORATIVE CONTENT

Web 2.0 technologies, discussed in various parts of this text, have enabled many powerful collaborative content management and social networking tools. Blogging, news feeds and aggregators, podcasting, tweeting, and collaborative content management with wikis have been added to the earlier Internet tools of email discussion lists and Usenet groups. Although content generated by some of these communication tools, such as IM and chat, tends to be somewhat ephemeral, others enable collaborative content production that results in new sources of searchable information accessible by both the participants and the public.

Email Discussion Lists and Usenet

Email discussion lists were early collaborative content tools of the Internet and are still extensively used in many organizations. Programs such as List-serv or Majordomo can create group lists, manage subscriptions, and archive list content. Discussion lists exist around many topics and specialties in the library profession, and they serve as an important repository of human expertise and problem solving on a wide range of topics. Usenet groups are another form of a virtual community that grew out of the availability of Internet email. Instead of having messages "pushed" to each member as in discussion lists, users post messages to a common area where others can read them or reply. These groups are similar to the old "computer bulletin boards" that were common in the pre-Web computer era. Usenet groups represent every imaginable interest, and extensive lists of these groups can be found on the Web. For instance, Google Groups lists thousands of groups on a huge range of topics. Much of this content is archived and searchable if desired; some can be subscribed to with RSS feeds much as with a blog.

Blogs, News Feeds, Tweets, and Podcasts

A huge source of new content comes from the proliferation of blogs, news feeds, tweets, and podcasts now common on the Internet. The foundation technology for many of these tools is the RSS protocol, an XML-based standard. Blogs are a Web-published narrative of an individual or group, a shared “Weblog,” that is updated regularly. The blogger frames the discussion by initiating the threads and determines if, and how, reader comments can be added. Some blogs are very personal narratives, and others may focus on a particular subject area. Blogs have some of the feel of both Usenet and discussion lists, but there are several important distinctions. They are much more interactive, and they reach a much wider audience. It is likely that many people who never participated in Usenet groups or email discussion groups have now discovered the blogosphere; a 2010 Pew Internet and American Life Project survey estimated that in 2009, 15 percent of adults and 14 percent of teens were bloggers and many more may just read them (Lenhart et al., 2010). In addition, most traditional news sites offer blog subscriptions to their readers. Blogs now receive much media attention, and the buzz generated on a blog increases the probability of the story being picked up by mainstream news outlets. The *Drudge Report*, founded by blogger Matt Drudge, is a news aggregator that began as a simple email list but has become a news source that is now available to mobile devices (Wellman, 2007). What began as just a personal, but public, journal for individuals is now a successful information outlet that competes with traditional media.

The software to create and manage blogs can be installed locally or accessed on a Web hosting service, such as Bloglines (<http://www.bloglines.com/>) or Blogger.com (<http://www.blogger.com/start>). Local reader software or reader websites are also available for people following syndicated blog content. Some library blog topics include reference “frequently asked questions,” general announcements, new books, book clubs, and specialized areas such as business or health sources. Blogs are also used as a collaborative content management approach to create home pages.

Although still popular, the number of bloggers has decreased somewhat while the number of Twitter users has grown, especially among the younger users of social media. Twitter is now a huge source of news and information, and the tweets of its users contribute vast amounts of content even though individual tweets are limited to 140 characters. As of 2013, some 16 percent of online users use Twitter, and the number that use it every day has quadrupled since 2010 (Duggan & Brenner, 2013). Libraries have found many ways to use both blogs and Twitter to share and push content both internally and to the broader community.

Podcasts represent another important content stream. Podcasts are digital audio programs delivered using RSS and can be downloaded to MP3 players; they can be described as audio blogs. Not only is it easy to find and hear podcasts, but with a few simple tools, anyone can become a producer of a digital radio show. Podcasts are also showing up in many library venues, replacing cassette tape audio tours and recorded stories for kids. In many ways, podcasts and the blogosphere have become the new form of “talk radio” where anyone can reach a mass audience on any topic.

Wikis

The wiki, introduced in Chapter 2, is software that allows anyone to create Web content collaboratively with others without having to know HTML. They facilitate crowd sourcing by allowing anyone to share their expertise on a topic. Wikis have become an important source of Internet-based reference, such as that found at wikipedia.com, but some controversy surrounds wiki use as a reference tool because of the lack of quality control mechanisms. There are documented instances of malicious or biased editing in Wikipedia, but there is also evidence that its accuracy compares favorably with traditional encyclopedias (Giles, 2005). Aside from reference, Wikis present new opportunities to build and share content in many forums, and their use is growing, for example, my university uses wikis to support IT and the Blackboard course management system. Some examples of these applications in libraries are discussed further in Chapter 16.

MULTIMEDIA ON THE WEB

One of the powerful and engaging aspects of the Web is the capability to embed multimedia elements into websites. Not only does this provide for entertainment, but it is also a powerful educational tool because different learning styles can benefit from alternate views of textual content. The vast number of mobile devices in use has made posting video to social media sites or YouTube extremely simple and has led to an exponential increase in such content. However, the ease with which archival multimedia can be presented on the Web depends in part on whether the content was “born digital”; analog formats must be converted to digital, which can involve a significant effort. There are a number of technologies and formats used for Web multimedia.

Creating Sound and Video Files

Conversion of analog sound to digital signals requires sampling the wave amplitude with an Analog to Digital Converter (ADC); and playback of digital audio requires a complementary Digital to Analog Converter (DAC) to recreate the sound waves people hear. As discussed in Chapters 1 and 7, sampling rates determine the quality of the conversion. Sampling technologies are based on the Nyquist–Shannon Sampling Theorem or its modern variants (A new wave, 2002). For instance, conversion of analog music to CDROM typically uses a sampling rate of 44.1 kHz, which is sufficient to represent the analog version accurately (Watkinson, 2000).

The capability for this type of conversion is available on almost all PCs and many mobile devices. Audio destined for Web delivery could come from a variety of media sources, such as CDROM, audio tape, vinyl record albums, or MP3 files. Audio in MP3 is already Web ready, and CDROM sources are easily ripped to MP3 format by many programs such as iTunes or Windows Media

Player. Audio tape and vinyl records take more effort, but there are devices for each type of media that can be connected to a PC via a USB port.

Much video is now born digital, and video capture technologies are available for converting older analog video formats such as VHS tape to digital. Once again, the amount of sampling determines how accurately the digital version represents the full frequency ranges of the original analog form, which in turn determines final file size. The availability of newer digital recording devices has made it easier to work with both audio and video content directly on the PC.

In general, digital formats for multimedia are designed with a variety of compression strategies to keep file sizes as small as possible. For instance, when recording an audio lecture, much of the recording is actually saving “silence” (the pauses between words), and algorithms compress the amount of digital information needed to represent those background periods of silence. The information captured for video can be thought of as a form of “3D”; video is a series of two-dimensional pictures where the third dimension is time. Because much of the image captured over this timeline is background that does not change from frame to frame, programs need only focus on the information that updates what is different in the new frame from the previous one. When enough of the frame is different from the previous one, a new “base frame” is used for this updating process. Production values for Web-delivered video are an important consideration, and a video professional can advise about the studio environment, appropriate lighting, and stage directions because excess movement of the participant can affect the quality of final product. In addition, bandwidth issues associated with different display preferences should be considered.

Because of the compression and decompression needed for creation and playback, video files are more complicated than many other formats. The structure of the file is determined by its container, and its codec controls its compression and playback. Different codecs are available in the form of software and hardware; for example, Microsoft has proprietary codecs for its WMV and MS-MPEG (Motion Picture Experts Group) formats. The H.264/MPEG-4 codec (also called Advance Video Coding or AVC) is the standard employed for most high-quality video. The sampling bit rate and the codec in use determine the quality of the playback.

Even with high-compression formats, digitized video can require huge amounts of data to be stored. Fortunately, disk storage has become cheap, and many terabytes of storage for video can be quite affordable. Bandwidth is critical for the user viewing experience, and a high data transfer rate must be sustained to avoid “jerkiness” in the playback. The older VHS video standard displays 30 frames per second, which gives the illusion of smooth, continuous motion. The amount of digital data that must be captured depends on both the frame rate chosen and the size of the video image on the screen. Early PC video formats typically used lower frame rates and did not utilize the full screen; the early Apple QuickTime format used 15 frames per second and often displayed video in a small window. The AVI (audio visual interleave) format was developed in 1992 by Microsoft for Video for Windows. Another format is AMV video (also called MTV), a proprietary format based on AVI. The large file sizes associated

with many video formats create a bandwidth problem for some users, especially those with slower connections; any video streaming is problematic without broadband connectivity.

Fortunately, there are technologies and formats that make multimedia much more Web friendly. The MPEG standard is a lossy compression method that has evolved into multiple formats used for high-quality audio and video such as MPEG-2 for DVD movies. MPEG-3 (usually now abbreviated as MP3) was combined with MPEG-2 and compresses CD-quality sound by a factor of 12 while maintaining high fidelity; it is the format responsible for the success of MP3 players. MPEG-4 (MP4) is the latest iteration of this format designed specifically for digital video and video streams.

Streaming Audio and Video

Streaming technology has made video on demand a popular service and a missed television show or movie is available from many sources including the networks themselves or sites such as Hulu, Netflix, or Amazon Prime; by 2012, Netflix streaming topped a billion hours per month (King, 2012). For both audio and video, streaming technologies allow a bandwidth-dependent data stream to take place between client and server. Prior to this approach, a media file had to be downloaded in its entirety before playback could begin, which resulted in long delays before the user could even see if the media was of interest. Media can still be streamed on the Web with HTTP simply by uploading the media file and making a link to it or embedding it into a page. Playback is then dependent on the browser opening an appropriate media player. Such an approach is fine for low traffic video, but for high-traffic sites a dedicated streaming server is needed that allows dynamic playback that can match bandwidth to the data being delivered. Streaming servers typically employ the RTSP (Real Time Streaming Protocol), which in turn uses RTP for packet formation. It establishes and controls media sessions between client and server. Examples include the Real Networks Helix Universal Server, the Apple Quicktime Streaming Server, and the Adobe Media Server.

Streaming media presents challenges to content producers in that there are many format options and many players for them. For example, online classes using the Blackboard course management system might provide a great deal of video content, but its use is potentially limited to certain devices or browser plugins. One solution for managing video content is the CMS-like open-source Kaltura video platform, described as “the world’s first and only Open Source Online Video Platform” and “a framework for developing custom applications” (About Kaltura, 2013). Kaltura is available as a hosted “software-as-a-service” or a version that can be installed on local servers. Kaltura can handle many of the media types, such as mp3, m4v, mp4, mov, flv, wmv, wma, or avi, described in the next section and then convert them into file types suitable for the Web. As with other streaming technologies, it can adjust playback to the available bandwidth. For example, a FLV file could be transcoded into multiple MP4 versions to provide optimal playback for different users. A Kaltura Building Block is available as a commercial application and the Kaltura player enables video playback anywhere across multiple platforms and OSs (Video building block, 2011).

Common Multimedia Formats

Some of the multimedia formats commonly encountered on the Web are described below:

- **RA, RV, and RAM files:** These are various formats developed by Real Media. The **RA** format is audio, and **RV** is video with or without audio. **RAM** files stand for “Real Audio Metadata” and are text files where the browser receives the RAM file, launches Real Player, and then requests the various content sources listed as separate URLs in the RAM file from a Real Server.
- **SMIL** (Synchronized Multimedia Integration Language) files: An XML-based markup language being developed by W3C that enables Web producers to divide content into separate files and streams but display them together.
- **SWF**, or Shockwave Flash file: A proprietary format developed by Macromedia, Inc. and now owned by Adobe, Inc. This format, which uses vector objects for images, was developed for delivery of simple multimedia animations on the Web. It required the installation of a free browser plug-in for display, but most users have, or can easily get, the Adobe Flash player if their device and its OS support it. Flash has been an important format for content delivery but its use is declining (Usage of Flash for websites, 2013). One application of this format was as an output format for the Camtasia Studio program to convert PowerPoint slide shows with accompanying audio to the Flash format for use in online courses. The resulting package was quite compact. For example, a 30-minute PowerPoint lecture output as Flash often yielded a SWF file that is only about 5–12 MB. However, SWF files can be resource demanding and not as compact as newer format options. Flash use on websites has declined from about 24 percent to about 18 percent during the 12-month period that ended in July 2013. In 2010, Apple created some controversy when it announced it would not support Flash on its iOS, but it did eventually allow third-party Flash players to run on their systems. Adobe will not continue development of the Flash player for mobile beyond Android and Playbook devices (Bell, 2011). The newer versions of Camtasia have dropped support of Flash as a Web output format and recommends MP4 in its place (O'Rourke, 2012).
- **FLV** or Flash Video File: A container file format; it is a Flash format that followed the SWF type (Fisher, 2013). It is extremely common, especially on video sharing sites such as YouTube where it is a de facto standard. It is designed for use with the Adobe Flash player but other players work with it including the RealPlayer. It contains a proprietary video bit stream derived from the H.263 video compression standard delivered to the player via HTTP as a progressive download. Although such a strategy does not dynamically respond to bandwidth, it uses buffering to allow playback to begin before the download is complete. FLV has some performance and support issues similar to those of SWF, and it appears to be declining as more developers turn to MPEG4 (Meadows, 2013).

- **AVI**, for Audio Video Interleave: This is an older proprietary video format from Microsoft that has been replaced by the newer WMV format. Since its development, many advances in video technologies are not supported by AVI. For example, it does not support streaming or the H.264 codec used by MP4 and MOV (Difference between MOV and AVI, 2013).
- **WMV**, for Windows Media File: Microsoft's improved media format that uses various proprietary codecs. It supports streaming and could thus compete with other streaming options such as Real Video.
- **MOV**: The format for Apple's Quicktime video player, it is a proprietary format that uses lossy compression to reduce file size (Difference between MOV and MP4, 2013). Because it became the basis for the later MP4 file standard, these are quite similar and the choice between them is therefore determined by the expected audience preference.
- **MP3**: This audio format, described earlier in this chapter, is based on the combined MPEG-1 and MPEG-2 Audio Layer 3 standard from the Moving Picture Experts Group. It is a lossy compression format designed specifically for audio to achieve reasonable fidelity with much smaller file size than the corresponding CD recording; final quality is determined by the bit rate selected when creating the MP3 file.
- **MP4**: The extension for MPEG-4 Part 14 video files is also referred to as MPEG-4 version 2. Related extensions for MP4 include .m4a for files with audio only and .m4p for files protected with digital rights management technology.

Some common multimedia formats are listed in Table 13.2.

TABLE 13.2 Some Common Multimedia Formats

Extension	Format
AU	Unix audio file
WAV	Windows audio—wave file
AVI	Audio/Video Interleave—Windows
MOV	AppleQuickTime
RA	RealAudio
RM	RealMedia
RAM	Real Audio Metadata
RPM	Playback in a Web Page
SWF	Adobe Flash
FLV	Flash Video File
MP3	MPEG-1 or MPEG-2 Audio Layer 3; a patented encoding format for digital audio
MP4	MPEG-4 Part 14; a multimedia format typically for video and audio

COMPRESSION AND ENCODING FORMATS

Most multimedia files automatically use some level of compression as part of the format. However, compression in and of itself is also useful to minimize file sizes or combine multiple files into a single package for downloads or data packaging for sharing via the Internet. There are a number of formats specifically designed to accomplish such compression. As with many proprietary formats, there is a tradeoff involved, namely, both the sender and the recipient of the files need specialized software in order to compress and then subsequently uncompress them. A well-known compression file format for the PC is the **ZIP** format, which can create highly compact archive files. ZIP files require separate software, but the Windows platform is now able to create and use ZIP archives. There are also self-extracting **EXE** files that automatically uncompress when run. There are formats specific to other platforms as well, including UNIX **TAR** (tape archive) files and the Stuffit format for Apple. However, these compressed file formats can also hide malware such as a virus, worm, or Trojan program within a package file that can then be transmitted as an email attachment. This is another reason it is imperative to have antivirus software that can examine compressed files for embedded malicious programs.

Related topics to this discussion are the encoding schemes used to facilitate the movement of binary data via the text-based protocols of the Internet. Encoding schemes create a text representation of binary data and then convert it back to binary on the receiving end. Examples include UUEncode and UUDecode (uue files), BinHex (an Apple format that converts binary file into a hexadecimal format), and MIME, a scheme used by most email client software that automates the process of encoding and decoding of binary data.

INTERNET RESOURCE EVALUATION

Promoting information literacy has become a much-discussed role in librarianship. The American Library Association (ALA) defines information literacy as the set of skills and abilities to “recognize when information is needed and have the ability to locate, evaluate, and use effectively the needed information” (ALA’s Presidential Committee on Information Literacy, 2006). Evaluation of information is a key component of information literacy. Information professionals must assist their clientele in assessing the appropriateness and validity of the information they find in the Web environment. The development and broad acceptance of the Internet as a primary information source by both the public and librarians has made the issue of evaluation a critical one. Access and use of the Internet has grown significantly over the last several decades; nationally, nearly all colleges and K-12 schools have access to the Internet (Wells, Lewis, & Greene, 2006). In 1997, 79 percent of public libraries had access; by the end of the 1990s that grew to 95.6 percent; and by 2009, virtually all public libraries offer free Internet access (Public libraries and the Internet, 2009).

The issue of evaluation is confounded by the lack of any overarching control of Internet content, the malleability of digital content, and the ease and speed

of Web publishing. In traditional publishing, a number of gatekeepers are involved in selecting and vetting content. Publishers have a stake in protecting their reputation, and they employ multiple strategies to ensure quality, including peer reviews of work prior to its acceptance and reviews by editors who examine content for both factual errors and overall quality. In addition, the cost of production and marketplace dynamics combine to limit the world of printed output. The print-dominated world also had self-published fringe literature, but the cost and effort of self-publishing and the accompanying distribution challenges limited its dissemination. In the online world, anyone and everyone with a Twitter feed, blog, or website can easily become a content producer with a worldwide audience. The potential for misinformation, intentional or not, is great. Social media forums have created new avenues for news but lack the established checks for accuracy expected of traditional venues, and the “buzz” generated by a controversial blog report often becomes the story in and of itself, regardless of its merit. The democratization of content production is a two-edged sword; the Internet has been a great leveler of media, creating new opportunities for those who could not get access to a broader audience. However, when everyone can be a journalist or can present themselves as an expert, it puts the burden of ascertaining the validity of their content on the consumer.

There is a large amount of marginally useful or incorrect material on the Internet. The early Web enabled “vanity publishing” relating to vacations, pets, or other personal pages, but there were the barriers of technical skills and server space. Web 2.0 and the social networking phenomenon has removed many of these barriers and vastly increased the use of the Internet for personal narratives. There are significant issues posed by inappropriate and problematic materials such as pornography and hate literature. Some of these materials might be illegal; others may be offensive to many users but qualify as constitutionally protected speech. This has generated ongoing debates regarding government legislation and library policies designed to define and address this problem. Balancing how to best protect children from inappropriate materials while not concurrently imposing limits on free access to information continues to be a challenge.

A Pew Report revealed that by more than a decade ago, college-age students were using the Internet as their primary information resource and they reported little or no reason to use libraries or information professionals in their information seeking (Jones & Madden, 2002). As this trend continues, the concern of many in the library profession is not simply the competition the Internet presents, but the likelihood that people may be depending on, at best, poor quality unauthenticated resources or, at worst, dangerously inaccurate ones. In addition, there is the added “halo” effect that some Internet information seems to carry: the belief by some that if they find it on the Web, it is accurate and true.

Resource evaluation includes both an assessment of the content as well as consideration of the page design and presentation. Both can be evaluated, but clearly content assessment should be the higher priority. A number of framing questions facilitate the process of content assessment, such as:

- What is the apparent purpose of the site?
- What is the authority of the resource? For instance, is the author well known or clearly identified? Is the information verifiable in other

sources? How objective is the perspective? Is there an obvious agenda or bias?

- What about the publisher of the content—is it clear who sponsors the page? What is the domain location of site (gov, edu, com, etc.)?
- How complete is the information? Does it represent original content or just point to other sites?
- How current is the site? When was it created or last modified?
- What is the apparent audience level?
- Is there evidence of scholarship, such as citations that can be checked?
- Has the page been reviewed or recognized as important by others?

Information professionals have developed a number of educational strategies aimed at different audience levels to formalize critical thinking skills as applied to Internet resources. School media specialists have been particularly active in this area, striving to instill critical thinking early on with their students. Rubrics have been developed to help students structure the evaluation process. One example is “The five W’s” from Schrock (1998): Who wrote it? What does it say? When was it created or modified? Where does the information come from? Why is it useful? Another example is known by the acronym CARS: Credibility, Accuracy, Reasonableness, Support (Harris, 2007). Both of these simple rubrics can help students begin to develop a lifelong habit of critically evaluating Web content.

Although secondary to content, design also factors into evaluation. The best content is not very useful if presented in an unreadable style or without good navigation. Some of the key aspects to consider regarding design issues are:

- Graphics: Are they appropriate to the design or do they get in the way? Are the images sized appropriately? Do they load in a reasonable amount of time?
- Readability: Is the font size and style appropriate? Is there adequate contrast with the background color?
- Is there good navigation? Is it easy to get to the home base of site? Is it easy to navigate from each page to other major parts of the site?
- Is there a logical division and organization of the content?
- Is there good use of hypertext?
- Has access for the disabled been considered, such as with the use of alternate text for images?

Evaluation of Web and Internet resources is a central component of the information literacy skills needed by information seekers in the twenty-first century.

SUMMARY

The Internet has greatly enhanced both the amount of information available and the ease of access to it. Internet information sources can be divided

into two primary types: content that is dynamically generated and static content found in autonomous file containers on a Web server. The focus of this chapter is not only the various forms of static Internet content found in the common documentary and multimedia formats but also the various interactive formats such as blogs, wikis, and Tweets that have added many new information sources to consider. The choice of a content format can become a barrier to information use, so Web publishers must balance the advantages and limitations each format presents. This is especially the case with multimedia content, which makes video platform solutions such as Kaltura an appealing strategy for managing and delivering media content. A common goal of Web publishing is to provide useful information that is accurate, well designed, and free of format or accessibility barriers to some community. Information consumers often must sort through many sources and critically evaluate each. Information literacy skills are essential to this process and information professionals are engaged in all these activities both as producers of original content and by assisting their clients to find and evaluate useful resources.

REFERENCES

- A new wave. (2002). *The Economist*, 362(8256), 68.
- About Kaltura. (2012). Retrieved July 17, 2013, from <http://corp.kaltura.com/>
- About-Kaltura.
- Berlot, J. C., McClure, C. R., Wright, C. B., Jensen, E., & Thomas, S. (2009). Public libraries and the Internet 2009: Study results and findings. Retrieved May 14, 2014, from http://www.ii.fsu.edu/content/download/16874/109694/03_executive%20summary1-7.pdf.
- American Library Association's (ALA) Presidential Committee on Information Literacy. (2006, July 24). Final report. Retrieved May 18, 2008, from <http://www.ala.org/ala/acrl/acrlpubs/whitepapers/presidential.cfm>.
- Bell, Killian. (2011, November 9). Adobe announces the end of Flash player for mobile devices. Retrieved July 16, 2013, from <http://www.technobuffalo.com/2011/11/09/adobe-announces-the-end-of-flash-player-for-mobile-devices/>
- Difference between MOV and MP4. (2013). Retrieved July 17, 2013, from <http://www.differencebetween.net/technology/difference-between-mov-and-mp4/>
- Difference between MOV vs AVI. (2013). Retrieved July 17, 2013, from <http://www.differencebetween.net/technology/difference-between-mov-vs-avi/>
- eBook formats. (2014). Retrieved April 18, 2014, from <http://ebookarchitects.com/learn-about-ebooks/formats/>
- Duggan, M., & Brenner, J. (2013, February 14). The demographics of social media users — 2012. Retrieved May 19, 2014, from <http://www.pewinternet.org/2013/02/14/the-demographics-of-social-media-users-2012/>
- Fisher, Tim. (2013). What is an FLV file? Retrieved July 16, 2013, from <http://pcsupport.about.com/od/fileextensions/f/flv-file.htm>.
- Giles, J. (2005). Internet encyclopedias go head to head. *Nature*, 438, 900–901.
- Harris, R. (2007, June 15). Evaluating Internet research sources. Retrieved October 1, 2007, from <http://www.virtualsalt.com/evalu8it.htm>.
- Jones, S., & Madden, M. (2002, September 15). The Internet goes to college. *Internet and American Life Project*. Retrieved October, 2007, from http://www.pewinternet.org/pdfs/Pip_College_Report.pdf.

- King, R. (2012, July 5). Netflix streaming tops 1 billion hours in month for first time. Retrieved September 8, 2013, from http://news.cnet.com/8301-1023_3-57467191-93/netflix-streaming-tops-1-billion-hours-in-month-for-first-time/
- Lenhart, A., & Fox, S. (2006, July 19). Bloggers: A portrait of the Internet's new storytellers. *Internet and American Life*. Retrieved October 1, 2007, from <http://www.pewinternet.org/pdfs/PIP%20Bloggers%20Report%20July%2019%202006.pdf>.
- Meadows, Chris. (2013). A look at the FLV file. Retrieved July 16, 2013, from <http://pc.answers.com/computer-science/a-look-at-the-flv-file>.
- O'Rourke, Dave. (2012). Camtasia Studio support of MP4, SWF, and FLV file formats. Retrieved July 16, 2013, from http://feedback.techsmith.com/techsmith/topics/camtasia_studio_support_of_mp4_swf_and_flv_file_formats.
- Schrock, K. (1998). 5 W's for evaluating Web sites. Retrieved October 1, 2007, from <http://kathyschrock.net/abceval/5ws.htm>.
- Usage of Flash for websites. (2013, July). Retrieved July 16, 2013, from <http://w3techs.com/technologies/details/cp-flash/all/all>.
- Video building block for Blackboard 9.x. (2011) Retrieved July 17, 2013, from <http://exchange.kaltura.com/content/video-building-block-blackboard-9x>.
- Watkinson, J. (2000). *The art of digital audio* (3rd ed.). Oxford: Focal Press.
- Wellman, S. (2007, December 5). Drudge report goes mobile. Retrieved May 19, 2008, from http://www.informationweek.com/blog/main/archives/2007/12/drudge_report_g.html.
- Wells, J., Lewis, L., & Greene, B. (2006). *Internet access in U.S. public schools and classrooms: 1994–2005*. Washington, D.C.: U.S. Department of Education.

WEBSITES OF INTEREST

- RealNetworks Helix at <http://www.relnetworks.com/helix/index.aspx>.
- Apple Quicktime at <http://www.apple.com/quicktime/extending/resources.html>.
- ALA recommended sites at <http://www.ala.org/parentspage/greatsites/criteria.html>.
- Evaluation tools at <http://discoveryschool.com/schrockguide/eval.html>.

Information Retrieval

The goal of this chapter is to condense the essential concepts and classical models of information retrieval (IR) into a summary overview necessary for the discussion of Internet search engines in Chapter 15. Battelle (2005) describes Internet search as the most transformative technology since the development of the PC in the 1980s, and the IR models and strategies unique to its development are the focus of Chapter 15. However, Internet search technologies are an application of the general principles of IR, so knowledge of IR principles, performance measures, and classic IR models is a prerequisite to that discussion. IR is a complex and diverse topic, and its coverage here is admittedly superficial and presumes some familiarity with these topics. Those who have not yet had such exposure to IR or who wish to explore it in more depth should consult one of the many excellent texts referenced throughout this chapter.

INFORMATION RETRIEVAL OVERVIEW

IR is a broad topic covered in many different texts (Baeza-Yates & Ribeiro-Neto, 1999, 2011; Chu, 2003, 2010; Korfshage, 1997). A good starting point for this overview is to begin with a few definitions of IR provided in these and other sources. Bawden (2007) interprets IR as a process, describing it as “the purposeful searching for information in a system, of whatever kind, in which information—whether in the form of documents, or their surrogates, or factual material ('information itself'), are stored and represented” (p. 126). Baeza-Yates and Ribeiro-Neto (1999) describe IR as the “part of computer science which studies the retrieval of information (not data) from a collection of written documents” that “aim at satisfying a *user information need* usually expressed in natural language”

(p. 444). Korfhage (1997) provides a definition of IR particularly suited to the focus of this discussion, describing IR as “the location and presentation to a user of information relevant to an information need as expressed by a query” (p. 324). Essentially, IR systems function as an intermediary between information resources and a user’s information need, as shown in Figure 14.1.

Definitions of IR therefore depend on a number of associated concepts, specifically the nature of *information*, how it is represented and stored, and how it is sought and selected from the potentially large amount of other, nonrelevant information in the system. Defining “information” is the first challenge faced in any exploration of IR. This common sense concept is surprisingly difficult to define, because it has many different possible meanings in different contexts. How is information the same or different from the related concepts of data, knowledge, or even wisdom? It is intuitively understood that words, images, music, and pictures all represent forms of information, but so do aromas, the color of a threatening sky, and facial expressions. A broad summary definition from Case (2007) is that “*information* can be any *difference* you perceive, in your environment or within yourself. It is any aspect that you notice in the pattern of reality” (p. 5, emphasis by the author). A more limited view of information emphasized in this discussion leading to Internet searching considers information to usually be in the form of recorded textual or multimedia information that can be processed and represented in a system for subsequent retrieval.

Calvin Mooers introduced the term *information retrieval* in his 1948 master’s thesis at MIT (Morville, 2005). Since that time, IR systems have utilized various strategies to represent, store, and facilitate retrieval of information by accepting and processing queries that represent some information need. The information is typically in the form of documentary resources as represented by sets of symbols that can be manipulated by both people and computer programs and that have meaning to the user. Textual information uses symbols such as letters and digits that are on a page or in digital systems, magnetic charges on a disk, or some other electronic representation. One of the first IR technologies used punch cards with notches for index terms; these cards were physically manipulated to segregate matching cards from the rest of the set. Cards representing nonrelevant retrievals that fell out with the relevant ones were called *false drops*, a term still encountered in IR to describe nonrelevant retrievals returned in a search.

Database management systems often use a command-based query language to extract records that match some query parameters. Relational databases are

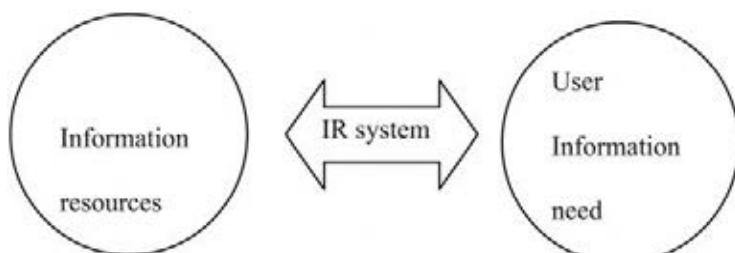


Figure 14.1 IR systems serve as an intermediary between information resources and someone with an information need.

one type of database system and are briefly discussed in Chapter 11 in conjunction with dynamic database-driven websites and CMSs. Relational databases use a command language known as SQL, often in the form of a “select” statement. While the backend processing of search engines may utilize SQL, the user interface typically allows open-ended keyword searching that does not require the user to utilize its formal command-based syntax. Websites or search engines that deliver information from databases typically utilize scripts to handle that part of the query process transparently. Database management systems and their specific query languages are an important area in the field of IR but are outside the scope of this book.

TEXTUAL IR SYSTEMS

Textual IR systems are the initial focus of this overview, and there are excellent sources that treat this area of IR in more depth (Meadow, Boyce, & Kraft, 2000). Beginning in 1990, the Text REtrieval Conference (TREC) has advanced many new technologies by providing a test bed for research and development activities in the field of textual IR. Internet search engines have adopted many of the developments and technologies that have resulted from TREC initiatives. The TREC has the following stated goals:

- To encourage research in IR based on large test collections;
- To increase communication among industry, academia, and government by creating an open forum for the exchange of research ideas;
- To speed the transfer of technology from research labs into commercial products by demonstrating substantial improvements in retrieval methodologies on real-world problems;
- To increase the availability of appropriate evaluation techniques for use by industry and academia, including development of new evaluation techniques more applicable to current systems. (The Text REtrieval Conference, 2008, para. 2)

In addition to supporting textual retrieval, TREC activities have diversified to include nontextual IR research, such as the track on retrieval from digital video launched in 2001 (Smeaton, 2007). Multimedia IR is briefly discussed later in this chapter.

In general, text-based retrieval begins with the presumption of some set of documents and user information needs that can be satisfied with the successful retrieval of one or more specific documents within the document corpus. The first challenge is to process and represent the documents in the IR system. Document representation can be accomplished through the creation of a surrogate, which can be a summary or abstract, a metadata record, or indexes of individual words or phrases either assigned to, or derived from, the document.

As the name implies, metadata is generally information that describes or defines some other information object, often according to a specific set of rules. Such a structured description of an information object can then

serve as a surrogate for that object in an IR system. One example of such a metadata-based document surrogate is the bibliographic record that is created and stored in library catalog databases. These records represent information objects (e.g., books) and provide access points to them that are used for their retrieval. The card catalog of days gone by is a classic example of a searchable collection of document surrogates. The bibliographic record on each card represented the item of interest. The duplication of cards by access point allowed searching by author, title, and subject, and each card for an item pointed to a physical location with a call number. Such highly organized and labor-intensive catalogs worked well with the relatively small and well-defined collections maintained by librarians dedicated to the task. Modern computer-based catalogs allow for the creation of indexes using all these access points as well as any keyword occurring in the surrogate record or the full text of the document itself. These index terms point to the surrogate record or document, which in turn points to the physical or digital item in a collection. Full-text digital resources can also be organized according to some classification scheme or utilize metadata to enhance intellectual access to them. As highlighted in the next chapter, the use of embedded metadata to support semantic search is becoming more common, but in general there is still little organizational structure or formal metadata available to facilitate access to most Internet resources. One of the benefits of the a future semantic Web based on XML, envisioned by some as “Web 3.0,” would be the opportunity for more effective metadata-enabled Web searching. Internet search engines use indexes for document representation in the form of an inverted file, as described next.

INDEXES AND INVERTED FILES

Indexes consist of an ordered list of terms that provides a link to the document, or documents, it represents. Index terms can be manually assigned by an indexer or generated automatically by extracting terms either from the documents or from their surrogates. These two approaches will be compared later. The importance and usefulness of an index, such as the one found at the end of a book, is well understood as a means of locating information. An index provides a list of terms, often arranged alphabetically, associated with record identifiers that facilitate quick lookups by common access points. Indexes are highly effective IR tools that eliminate the need for a sequential string search through a large set of records. For instance, in the library database example, separate indexes exist for author names, title words, subject heading terms, and a combined keyword index derived from all record fields. The index of a book is a collection of keywords showing the page numbers where the term occurs. The book example reveals some of the limitations of an index; the usefulness of the index depends the term of interest being included and on how well the term represents a concept. Computer technology and digital resources facilitate the automatic parsing of a document’s full text into one or more indexes. An index term can point to the set of documents that contain it, a record level index, or the specific documents with mapping of each term occurrence, which is a word-level index.

The net result of this process is the creation of an *inverted index*, sometimes referred to as an *inverted file index* or simply an *inverted file* (because the index itself may be a separate file in the system). The inverted index is defined as a data structure that provides a list of words to serve as access points into a set of texts. Each index entry is associated with the list of texts in which it occurs, and perhaps the location of the word within each of those individual texts (Black, 2008). The process of associating index terms with unique identifiers of specific objects is the process of inversion. When the locations of the terms within a document are also mapped, the process is referred to as *full inversion*. Although some sources distinguish an inverted index from an inverted file index (Black, 2006), for the purposes of this discussion, these are considered synonymous and will be used interchangeably. The inverted file is a well-known structure that facilitates quick lookups using common access points, and many IR systems and Internet search engines make use of them.

Flynn (1987) explains this structure with an example of a record level inverted file derived from a simple data file, and a similar example is provided here. The data file is about a wine collection and has information about the vineyard, the type of wine, the variety of grape, and the vintage year. Each record has a unique identifier, in this case a unique numeric key, assigned at the time of the record creation. A portion of this hypothetical database appears in Figure 14.2.

This simple example highlights two important points. First, the records must have unique identifiers assigned to them that can then be referenced later. Second, any field can be a candidate for inversion; any field that might be the source of a common query is often inverted. For instance, if a query seeks to list all red wines of a specific variety, the database engine could perform a sequential search of each full record looking for records that contain text string matches with the query. However, if this is a common query, a more efficient strategy is to utilize an inverted file structure. The result is a list of the possible values for the inverted field along with pointers to matching records. Inverting on the “Type” field yields the inverted file structure shown in Figure 14.3.

Inverted indexes can be constructed by human indexers or through automated document processing. The advantage of human indexing is that a person

ID	Vineyard	Type	Variety	Vintage
001	Yellowtail	Red	Merlot	2004
002	St Michele	Red	Cabernet Sauvignon	2006
003	Falkenburg	White	Riesling	2005
004	Little Boomey	Red	Shiraz	2006
005	Montes	White	Chardonnay	2004
006	Macmanis	Red	Pinot Noir	2002
007	Lancers	Rose	Portuguese	2005
008	Bonterra	White	Sauvignon Blanc	2006
009	Wolf Blass	Red	Cabernet Sauvignon	2000
010	Louis Jadot	Red	Beaujolais	2007

Figure 14.2 A portion of a database table.

Type	Records
Red	001, 002, 004, 006, 009, 010
Rose	007
White	003, 005, 008

Figure 14.3 Inversion on the field named “Type.”

analyzing the resource may assign terms not found in the document itself; the disadvantage is that indexing requires time and effort. Automatic document processing is more efficient but lacks this human subject analysis. Index terms can be combined or *pre-coordinated* into a meaningful phrase such as “Learning Disorders,” or terms can be *post-coordinated* by the user in the retrieval query. Automated indexing requires some form of document processing to create word-level indexes that are derived from full-text sources. In addition, pre-coordination of terms can be automatically achieved by examining term proximity and term co-occurrence within the document. The document is broken up into its component terms, stopwords are removed, and pointers are created for the set of documents that contain the term. *Stopwords* are words deemed meaningless as access points and include articles, prepositions, and conjunctions. Other information about the term may also be stored, such as the term’s frequency, its position in the document, the structural part of the document it came from, and the overall length of the document. Such information can be used to determine the *weighting* of the term by the retrieval system; for instance, a term that occurs multiple times in a document would be assigned a higher weight. This is the type of information typically associated with Internet search engines’ indexes.

The resulting index is comprised of vocabulary derived from the texts, and the term’s location in a text is associated with an address. Computer systems can keep track of the entire set of individual characters that make up a document, or they can treat the document as a group of logical blocks of text. Index term occurrences can therefore be based on their character positions, their location in some block of text, or the word position within the entire document. The tracking of terms in a document reflects the level of *addressing granularity*, which determines how term occurrences can be manipulated with proximity or phrase searching. Inverted file index structures are used with Internet search engines such as the Google index; the use of the inverted index in that context is discussed in the next chapter.

VOCABULARIES

When processing queries, an IR system must match query terms against the terms used to represent the document; both index creation and subsequent queries therefore depend on some vocabulary or lexicon. In the indexing process, term selection may utilize controlled or uncontrolled approaches. When terms are drawn from controlled vocabularies, they come from a restricted set

of predetermined words. In uncontrolled vocabularies, terms are unrestricted keywords, often derived from the natural language of the document itself. Each choice determines the appropriate lexicon of possible query terms. Indexing terms derived from unrestricted natural language may be presented to the system in a variety of ways. A natural language processing (NLP) search system implies that queries may be posed in a natural conversational fashion as opposed to a specified, structured format. The early Ask Jeeves search engine, now Ask.com, is an example of a Web search engine utilizing this conversational approach to query formation. Even as voice search systems such as Google's voice search and Apple's Siri have gotten very good at interpreting natural language queries, the ambiguity of natural language still poses many challenges to NLP systems. Not all keyword IR systems allow natural language queries. Term selection may be unrestricted, but query formation in many databases may require specific formatting, such as putting term phrases in quotation marks or adding an index name prefix or suffix to the query terms (e.g., "Bruce Springsteen" or "Wodehouse: author").

Controlled Vocabularies

A controlled vocabulary is an indexing language in which the semantics and syntax for query terms are formally defined and applied by human indexers. In this approach, the representation of the resource is restricted to a predefined set of words; both the index terms and successful queries must utilize the same defined lexicon. Often a thesaurus is available, guiding users to appropriate terms and illustrating term interrelationships. This structure leads searchers to similar, broader, narrower, and related terms as well as explaining the scope of the term. Controlled vocabularies are often associated with descriptor terms that reflect the intellectual content, but other controlled "authority lists" can be created to standardize names of people, places, or subjects.

Controlled vocabularies are used in many forms to represent the intellectual content of an information resource with descriptors, subject heading lists, or classification schemes. Examples of controlled vocabularies include Library of Congress Subject Headings (LCSH), the ERIC Descriptors for Educational Resources, developed by the U.S. Department of Education, and MeSH Medical Subject headings developed by the National Library of Medicine. Controlled vocabularies can enhance search precision and reduce term selection problems. In addition, controlled vocabularies facilitate exploratory searching by leading someone unfamiliar with a discipline to new terminology that could be useful in a search. However, these advantages come with a cost to both the indexer who must assign them and the searcher who must be familiar with the preferred vocabulary and its structure.

Because the stated purpose of this overview is to develop needed background for an examination of Internet search engines, it is reasonable to ask how controlled vocabularies relate to Internet search. The short answer is that they are not used for most Internet searching. In fact, if anything the trend seems to be moving away from controlled vocabularies approaches and toward user-generated tag clouds known as folksonomies. In the early 1990s, just as

the Web was about to burst on the scene, librarians were exploring cataloging of Internet resources; an OCLC study suggested full MARC cataloging could be applied to Internet content (Dillon, Jul, Burge, & Hickney, 1993). With the rapid growth of the Web, this quickly was recognized as untenable. However, new metadata schemes, such as DC, provide more streamlined approaches that, when combined with controlled vocabularies, could offer improved precision when used with “metadata aware” search engines. The promise of the XML-based semantic Web and the use of embedded metadata could result in a more structured Internet with additional opportunities for controlled vocabularies as well as making more granular connections among data from different resources.

Automatic Indexing

Automatic indexing uses the natural language of the documents themselves to represent them and permit queries with unrestricted term selection formed from a natural language vocabulary. A given document word set contains two subsets: (1) words that represent content, which are usually nouns and (2) words that are simply grammatical in nature, which do not convey meaningful content. Analysis of term frequencies reveal a relationship known as Zipf's law, which finds an inverse relationship between a term's rank and its frequency; so terms occurring with very high frequency have little to do with its content, and terms that occur few times have high ranks (Korfhage, 1997). As noted previously, automatic indexing begins with document processing to extract terms followed by the elimination of stopwords such as articles and prepositions, which have a solely grammatical function. Terms are sorted into an index that is wholly derived from the document lexicon itself. Document processing could extract terms only from specific locations in a document, such as a title, or from the full text. In addition, term weights can be calculated using parameters such as term frequency, its location in the document, its proximity to other terms, and overall document length.

Because automatic natural language indexing and searching is limited to the lexicon of the documents and the imagination of the searcher, it can result in document representation that is not as rich or precise as that achieved by a content expert using controlled vocabularies and a higher probability those queries may not use appropriate terms. However, automatic natural language indexes have several advantages: they are less costly to produce, and the user query terms are not restricted to a predefined vocabulary or descriptor language. Their hidden cost is a potential loss of search efficiency due to less precise search results and a higher proportion of nonrelevant retrievals. As described earlier, these nonrelevant retrievals in a retrieval set are *false hits* or *false drops*. The ambiguity of natural language makes the problem of false hits almost inevitable.

Query Formation

Queries formed with either controlled vocabularies and uncontrolled natural language are problematic due to *synonymy*, *polysemy*, and *homonymy*.

Deciding on the appropriate term to represent a concept or thing is a challenge because of the complexity and ambiguity of natural language. *Synonymy* refers to the fact that many words can represent a concept. Is it a car or an automobile, a film or a movie? A search system can include some term variants by truncation to a term stem, such as including singular and plural word forms, but synonyms are more difficult to address automatically. *Polysemy* refers to one word that may have related but quite distinct meanings depending on the context. The term “foot” could refer to anatomy or to the base of a mountain; “mouth” could be a body part or the end of a river; a “church” could be an institution or a physical building. *Homonyms* refer to words that are either pronounced or spelled the same but have distinct meanings. For instance, a “router” could be a network device or a woodworking tool; “lead” could be the metal, a person who leads, or the terminal end of a wire; “lie” could refer to an untruth, a verb for a physical action, or the position of a golf ball. Term ambiguity and subsequent term selection issues often result in the retrieval of many records that are nonrelevant to the query, as well as the nonretrieval of relevant ones.

To illustrate the usefulness of a controlled vocabulary in choosing query terms, an example is provided of a search of the ERIC database of education research. A hypothetical teacher interested in exploring the topic of Attention Deficit Disorders enters the query term “ADD.” The top retrievals this keyword search returns all have the word “add” in the title or abstract, but often in a different context. In the retrieved article shown in Figure 14.4 the word “add” appears not as the acronym ADD but in the sense of “adding” something.

The ERIC database has a controlled vocabulary documented in a thesaurus the searcher can reference. The thesaurus has an entry for “Attention Deficit Disorders” as a descriptor, as well as appropriate broader, narrower, and related terms shown in Figure 14.5.

Using the recommended descriptor in a new search results in a more precise retrieval set for the subject sought. The use of a controlled vocabulary, if available, is especially helpful in an initial exploratory search, which can lead to new terms for further searching. However, note the added precision comes with a cost to the searcher—they must not only know of the ERIC Thesaurus but also take the time and effort to consult it to locate the better term.

A top retrieval from ERIC using the keyword “ADD”
Course Shopping in Urban Community Colleges: An Analysis of Student Drop and Add Activities (EJ762247)
Author(s): Hagedorn, Linda Serra; Maxwell, William E.; Cypers, Scott; Moon, Hye Sun; Lester, Jaime
Source: Journal of Higher Education, v78 n4 p464-485 Jul-Aug 2007
Pub Date: 2007-00-00
Pub Type(s): Journal Articles; Reports - Evaluative
Peer-Reviewed: Yes
Descriptors: Community Colleges; Urban Schools; Course Selection (Students); Student Behavior; Two Year College Students

Figure 14.4 A retrieval from the ERIC database using the term “ADD.”

The ERIC Thesaurus entry for Attention Deficit Disorders
Attention Deficit Disorders
Descriptor Details
using Attention Deficit Disorders as a search criteria
Record Type: Main
Scope Note: Developmentally inappropriate inattention and impulsivity
Category: Disabilities
Broader Terms: Disabilities;
Narrower Terms: n/a
Related Terms: Attention; Attention Span; Behavior Disorders; Emotional Disturbances; Hyperactivity; Learning Disabilities; Neurological Impairments;
Used For: Methylphenidate; Ritalin;
Use Term: n/a
Use And: n/a
Add Date: 06/21/1983

Figure 14.5 The ERIC Thesaurus entry for Attention Deficit Disorders.

PERFORMANCE MEASURES

The goal of retrieval systems is to provide documents that are *relevant* to the query. People tend to think of relevance in a way that is analogous to the way the judicial system views the concept of obscenity, “you know it when you see it.” People evaluate a retrieval set based on an internal sense of whether the documents satisfy an information need; those that do are “relevant.” Automated ways to determine rank relevance are a significant challenge for IR systems; several algorithmic strategies for assessing relevance are examined with search engine technologies in the next chapter, including various forms of hypertext link analysis. Regardless of how relevance is determined, there are several standard measures of the effectiveness of a retrieval system: *recall*, the proportion of the potentially relevant documents that were actually retrieved and *precision*, the number of relevant documents within a given retrieval set. Recall is represented by the formula shown in Figure 14.6.

Applying this to an example, assume a database of 10,000 documents has 250 documents potentially relevant to some information need, for instance, information about breast cancer treatments. A search with perfect recall would then retrieve all 250 of these. If only 125 are retrieved, then recall is 125/250 or 50 percent. Note that this determination is problematic as it presumes one can know or estimate the total number of relevant documents that exist in the collection.

Precision measures the effectiveness of a search in terms of the proportion of relevant documents within all retrieved documents. Precision is represented by the formula shown in Figure 14.7.

$$R = \frac{\# \text{ of relevant documents retrieved}}{\text{Total } \# \text{ of relevant documents within the collection}}$$

Figure 14.6 Recall measure calculation.

$$R = \frac{\text{\# of relevant documents retrieved}}{\text{Total \# of documents retrieved}}$$

Figure 14.7 Precision measure calculation.

Continuing the previous example, assume the query “breast + cancer + treatment” retrieves 300 documents. If an evaluation of each in the set shows that 200 are relevant, then the precision of the search is calculated as $P = 200/300$ or 66 percent.

In an ideal world, searches would result in both perfect recall and precision. However, this is rarely the case because strategies to improve one measure usually reduce the efficiency of the other. For instance, an attempt to improve recall by “casting a wider net” through expanding the term set used in a search usually reduces precision by also gathering up a larger number of false hits. Conversely, attempts to improve precision by using more focused term selection usually results in lower recall.

Another highly important but more subjective measure of retrieval effectiveness is the time and effort required to process a query and display relevant results; both usability and search effort are critical factors. Clearly, a retrieval system that took many minutes to perform a search or that was difficult to use would not be acceptable to most users. The quantification of “too long” or “too difficult” is a judgment made in the context of the size and complexity of the corpus searched as well as the motivation and expertise of the searcher. The effort needed to use a system is inversely proportional to its acceptance by users. This was formally postulated by Zipf (1949) in his well-known “Principle of Least Effort” that postulated that “each individual will adopt a course of action that will involve the expenditure of the probably least average of his work” (p. 543)—that is, the least effort.

The truth of this principle is especially apparent in the preference of information seekers to use Internet search engines over more complex library systems. Even in the context of Internet searching, searchers seem to abandon complex or hard-to-use sites in favor of those that permit easy, fast keyword searches.

CLASSICAL IR MODELS

The three main classical IR models are the Boolean, vector space, and probabilistic models. These are not necessarily mutually exclusive, and each has multiple forms and variations; Internet search engine technologies utilize elements of all three models. In keeping with the overview goal of this chapter, the mathematical details and formulas associated with the vector space and probabilistic models are not discussed because the intent is to develop a “big picture” view of these topics to support an examination of Internet search in Chapter 15. In addition, this overview does not include the enhancements or alternatives that extend these models such as fuzzy Boolean, Latent Semantic Indexing (LSI), and the neural net model. More detail on these models is found in the references and suggested readings within this chapter.

Boolean IR

The Boolean model is based on set theory and Boolean algebra, and it is the most straightforward of the various retrieval models. Boolean logic has three operators: AND, OR, and NOT. IR systems utilizing Boolean logic permit you to “post-coordinate” the terms used in a query statement by connecting them together with any of these three operators. Boolean logic is represented by truth tables; numbers in these tables represent the presence or absence of a term and whether or not a record is retrieved. The generic example below shows the result for a two-term search using Boolean AND:

term1 AND term2 → only records with both terms are retrieved.

If “cats” and “dogs” are term 1 and term 2, a Boolean search for “cats AND dogs” only retrieves records that contain both terms. Boolean OR and NOT yield the results:

term1 OR term2 → Records with either one or both terms present are retrieved.

NOT term1 → Any record with term1 is eliminated from the retrieval set.

The Boolean algebra for these AND combinations is shown in Figure 14.8. The results of Boolean OR and Boolean NOT are shown in Figures 14.9 and 14.10.

Operand	Operator	Operand	Result
0	AND	0	0
0	AND	1	0
1	AND	0	0
1	AND	1	1

Figure 14.8 Boolean AND truth table.

Operand	Operator	Operand	Result
0	OR	0	0
0	OR	1	1
1	OR	0	1
1	OR	1	1

Figure 14.9 Boolean OR truth table.

Operand	Operator	Result
0	NOT	1
1	NOT	0

Figure 14.10 Boolean NOT truth table.

Boolean logic permits the formation of complex queries using combinations of these multiple operators, which follow established rules of operator precedence (NOT, AND, OR). Statements within parentheses are always processed first, so they are often used for complex Boolean statements to group expressions and control processing order.

The main limitation with standard Boolean retrieval is that there must be an exact match with the query terms. The Boolean model is binary in nature, that is, a document is relevant and retrieved or not retrieved depending on an exact string match. There is no “partial match” with simple Boolean; extended “fuzzy” Boolean does not have this limitation. The choice of query terms is therefore critical to successfully retrieving relevant documents, and the outcome of Boolean searching depends on how well the terms chosen to represent a concept match those in the index. All the issues discussed earlier regarding synonymy, polysemy, and homonymy are particularly relevant to Boolean searching.

There are several ways that a system could assist a searcher in identifying the best terms or extending the set of matches for the term used. Controlled vocabularies reduce term ambiguity and provide guidance on term selection with a thesaurus that identifies appropriate terms as well as term relationships. Even without a controlled vocabulary, the retrieval system may permit term truncation symbols that serve as placeholders for characters. The IR system could also automatically expand the term set used in the query by *term stemming* to take a word to its root form. Both these approaches add term variants to the query and expand the set of possible matches. In addition, search engines can provide term suggestions, auto-correct misspellings, auto-fill questions based on popular queries, and offer to search for results similar to a selected retrieval.

Vector Space Model

The vector space retrieval model described by Salton and McGill (1983) attempts to identify relevant documents through a mathematical computation of similarity between query and document vectors. This computation results in an estimate of relevance based on a computed degree of similarity measure, also referred to as the Retrieval Status Value (RSV). This approach avoids the binary limitations of the Boolean model and allows for system-generated term weighting. As the name implies, the calculation depends on creating document and query vectors and placing them in a multidimensional vector space for a similarity comparison. The similarity measure, typically the cosine of the angle formed between the two vectors, is calculated to determine relevance algorithmically. Higher cosine values mean smaller angles between the query and document vectors, which imply a higher relevance ranking; smaller cosine values mean less similarity between the query and document vectors and a lower relevance.

The process of term weighting is a highly useful feature of this model. To understand how term weights are generated, suppose a collection of documents ($d_1 \dots d_n$) has been processed and found to contain a set of unique terms numbered ($t_1 \dots t_l$), which therefore comprise the entire set of term vocabulary. Each term derived from the document set can be assigned a weight; this could

be a binary value (0 if absent, 1 if present) or, more likely, some intermediate weight value based on the term's frequency in a document. In addition to raw frequency measures, the uniqueness of the term can factor into its weighting as well; intuitively, it is logical to assume that terms occurring in many documents are less useful as descriptors than terms occurring in relatively few documents. This information is represented in a documents-to-terms matrix or array, as shown in Figure 14.11.

The matrix results in t -dimensional vector space, and because many specific documents would not contain many of the potential terms, the matrix would contain many zero values. Each term is an axis in a Cartesian coordinate space, and both documents and queries could be represented as vectors within the term vector space. A simplified view of such possible vector comparisons is in Figure 14.12.

	T_1	T_2	\dots	T_t
d_1	w_{11}	w_{12}	\dots	w_{1n}
d_2	w_{21}	w_{22}	\dots	w_{2n}
\dots				
d_n	w_{1n}	w_{2n}		w_{nn}

Figure 14.11 A document-term matrix. A set of n documents contains a set of t terms, each of which has been assigned a weight w , based on the term's frequency and other factors.

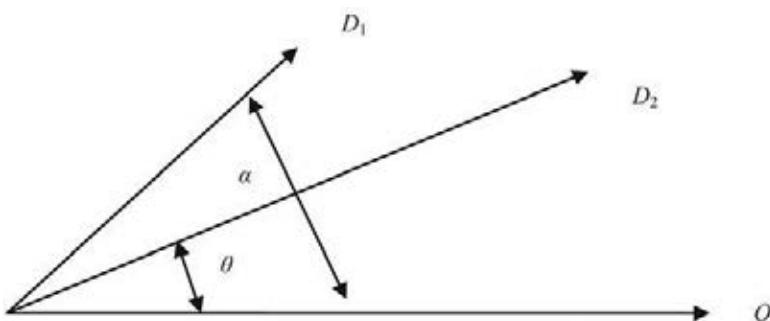


Figure 14.12 Document vectors D_1 and D_2 form angles α and θ , respectively, with query vector Q . The cosine of these angles will determine the degree of similarity; the angle formed with D_2 has a higher cosine value and is therefore more similar to the query than document D_1 .

This model has several distinct advantages: It permits partial matches based on index term and query word weighting and provides an algorithmic process to perform relevance ranking. However, one drawback is its computational complexity, especially in the context of a large-scale, heterogeneous document collection such as the Web. Computing term weighted vectors for every Web document followed by similarity computations with queries would be a daunting task for any system. However, there are techniques that make this approach more tenable for such a large corpus of documents; for instance, an initial term lookup in an inverted file can eliminate documents with none of the query words and greatly reduce the candidate set of documents for subsequent vector similarity calculations. The application of this approach to Web IR is discussed in the next chapter.

Probabilistic Model

The probabilistic model, also referred to as the *Binary Independence Retrieval* (BIR) model, uses a statistical approach to estimate a probability of relevance to a query. The model presumes some ideal retrieval set that perfectly satisfies a query and attempts to identify that set. The model uses binary term weighting and a recursive statistical technique to estimate the probability that a document is either relevant or not relevant to the query. The Google developers describe the PageRank algorithm as a recursive technique to develop a probability estimate (Brin & Page, 1998), so the notion of probability will come up again with the discussion of Google in Chapter 15.

NONTEXTUAL IR

Retrieving multimedia content presents many challenges to IR systems and their designers. Baeza-Yates and Ribeiro-Neto (2011) define the straightforward goal of these systems:

The task of a Multimedia IR system is to retrieve text, image, video, and sound data related to the interest of the user and run them according to a degree of relevance with regard to the user query. (p. 588)

However, the Web has seen a huge proliferation of such content and finding a specific media object has become more difficult as the quantity and diversity of these media have continued to grow. For instance, a common media type on the Web are images in various formats; by 2007 there were an estimated 10–20 billion images available to view and download, not including those in private photo collections stored online (Rorissa, 2007). Since that time, the proliferation of smartphones coupled with social media use has vastly increased those numbers; it is estimated that over 200,000 photos are uploaded to Facebook alone each day (Horaczek, 2013). Search options and retrieval success when seeking an image depends in part whether they are housed and maintained in structured databases or simply integrated within general Web pages. Textual IR dominates Web search, but Goodrum and Spink (2001) found searching for images comprises a significant portion of Internet information-seeking activity.

In addition, their analysis confirmed the difficulty of image searching for users of the Excite search engine at that time.

Users of specialized image databases may fare better given the nature of the image content and specific IR systems designed for those collections. Examples of such databases include the Library of Congress American Memory Project (<http://memory.loc.gov/ammem/index.html>), NASA's Multimedia Gallery (<http://www.nasa.gov/multimedia/index.html>), the National Institute of Health's Visible Human Project (http://www.nlm.nih.gov/research/visible/visible_human.html), as well as many art gallery databases. In addition to these projects, there are highly specialized IR systems designed for scientific and medical databases that contain mammogram or tissue-sampling images, or law enforcement databases for fingerprint or facial recognition (Cassetti, 2002). However, the large number of diverse image sources found on the broader Web can be more problematic for content-based pattern recognition programs. Other media types present their own set of challenges; for instance, content in a podcast is often in MP3 file format. Audio files and documents both contain textual content, one spoken, and the other written. However, the full text of the document is keyword searchable, but the podcast is not, unless a transcript accompanies it.

There are two main approaches to multimedia IR that can be used separately or in combination with each other. Cheng and Rasmussen (1999) describe these as *concept-based* and *content-based* IR approaches. Concept-based indexing and retrieval uses many of the same techniques employed for textual IR and involves the creation of a textual description for the media object that can be used by the system. The source of the textual description can be the object itself, the HTML code that references it or a more rigorous application of metadata by a human indexer or a program. Textual information can come from the object filename, a transcript, or a song's textual lyrics, or in the case of Web page images, from the alternate text or a caption. However, these sources of derived descriptive text are limited and sometimes not very useful; for instance, filenames may be nothing more than an automatic code and number such as "**DSC00268.jpg**." Alternate text or captions are often missing or applied inconsistently, resulting in poor, nonstandardized description. The creation of transcripts from audio content is not as easy as getting text from documents with OCR software, but there are voice recognition programs that do a reasonably good job of converting audio into text. The application of more rigorous standardized metadata can provide enhanced intellectual access, but it is a labor-intensive process. Alternatively, users themselves might contribute the descriptor tags they find meaningful and useful to multimedia. The large-scale Internet media search sites YouTube and Google Image search permit user tagging as a way to enhance access to visual content. Such a "tag cloud" of descriptors results in user-based taxonomies known as "folksonomies." Figure 14.13 shows an example of a tag cloud created by submitting the URL of the Librarian in Black blog to the tag generator Wordle.net.

The other main IR approach is Content-Based IR (CBIR), which uses programs that try to assess the actual content of the multimedia object in terms of color, form, or texture by examining pixels, waveforms, shapes, or other properties. The system then attempts to use pattern matching within the media



Figure 14.13 An example of a tag cloud where term frequency is visually displayed in larger fonts.

objects, for instance, to correlate image shapes with people, animals, structures, or other features to determine what it contains and what it is about.

The choice between these two strategies determines the query formation options. If concept-based textual indexing is used, the searcher encounters many of the same issues already discussed regarding term selection and natural language ambiguity. If a controlled vocabulary has been used the searcher needs to be aware of it to be directed to the appropriate terms. Content-based queries attempt to use features of the media object itself. This is done either in an iterative approach based on sample retrievals that allow the user to request more objects that look or sound similar to an initial retrieval, or by attempting to match user-specified patterns derived from colors, shapes, or sounds. These strategies are discussed further in the next sections.

Concept-Based Indexing: Textual Descriptors and Metadata for Multimedia

As noted previously, textual description can be derived from the content of the object itself, its caption, or associated alternate text. Deeper levels of indexing involve human intervention, but describing multimedia objects such as photographic images or music is subjective, and they are consequently difficult to describe consistently. “A picture is worth a thousand words” and “beauty is in the eye of the beholder” are two old adages relevant to this discussion. One of the major difficulties is that images are typically both *of* something and *about* something, and those might not be the same thing; this process is complicated further by cultural influences (Berinstein, 1999). A photograph of

a beach scene may include boats, waves crashing, sand dunes, people doing different things, and a beautiful sunset. The elements emphasized in the description of this landscape depend on what elements strike the viewer as important, and that assessment takes place at different levels because images are both literal and symbolic. For instance, a picture of a shrouded figure with a scythe could be described as a farmer in a hooded dark cloak, but many others would immediately recognize it as representing death. A picture of the planes hitting the Twin Towers is not just an image of a plane crash, but of a terrorist attack. Images, color, and symbols all reflect emotion and moods, and these second-level meanings are not the same for all viewers.

Textual descriptions for images and other media objects can be highly structured or relatively flexible and open-ended. On one end of the spectrum, there are established controlled vocabularies to assist in standardized description, such as the Getty Vocabularies that include The Art and Architecture Thesaurus, the Union List of Artist's Names, and The Getty Thesaurus of Geographic Names. At the other end, there are the user-driven folksonomies that comprise the uncontrolled vocabulary tag clouds assigned to images and videos on websites such as YouTube, Flickr, and Google Images.

Content-Based IR for Multimedia

An alternate approach to IR of multimedia is the development of programs that can “see” the image or “hear” the sounds in a way that is analogous to the way human sight and hearing match sensory data to conceptual models. Some technologies have been around for some time, for instance OCR programs for “reading” text as well as handwriting and speech recognition tools. However, because images are viewed at differing levels of abstraction, automated semantic representation is difficult. Programs can handle lower levels of abstraction, such as defining certain pixel areas as shapes that represent a tree or a person, but higher-level semantics is more challenging. Some of the more successful examples come from narrower applications, such as fingerprint image databases or medical imaging. Systems like the IBM “Query by Image Content” (QBIC) allow pattern matching based on image attributes such as shapes, colors, or textures. One example of QBIC color matching is used in a database at the State Hermitage Museum in St. Petersburg; users can choose colors from a palate, set the relative amounts of each color, and search their digitized art by these color patterns. The visual search company PiXlogic (<http://www.pixlogic.com/>) has developed iterative software that allows both shape and color pattern matching (Pepus, 2007). Another approach to image IR is “Query by Example” (QBE), where the system seeks more matches that are similar to a user-selected or generated example image. With this approach, the challenge is to locate an appropriate example as the starting point. These approaches can be combined; for example, QBE could be used to retrieve an initial image set and then QBIC applied to focus on certain shapes, colors, or textures, as the shopping site Like.com (<http://www.like.com>) attempts to do (Baxter & Anderson, 1996).

There are examples of very effective CBIR for music content such as the midomi service or app (<http://www.midomi.com>) and the shazam app—both allow you to hum a tune or record a music snippet and then use sophisticated

algorithms to search for a match in its database (Jacobs, 2010). Although not a direct matching service, both Pandora and Spotify use examples of your music preferences to attempt to find other similar artists and songs.

TYPES OF SEARCHING

Inherent in the definition of IR are the broad notions of information seeking and information behavior. Case (2007) describes information seeking as “a conscious effort to acquire information in response to a need or gap in your knowledge” and considers the importance of both active and passive information behaviors associated with it (p. 5). Although a broader discussion of this area is beyond the scope of this book, one facet of information seeking is particularly pertinent to the next chapter on Internet searching. Information searching behavior, described by Wilson (2000) as “the ‘micro-level’ of behavior employed by the searcher interacting with information systems of all kinds,” is relevant to Internet search (p. 49). Rosenfield and Morville (1998, pp. 102–103) describe four broad categories of information searching that influence the choice of the best search strategy:

1. The *known item search*, where a specific information item is sought. Examples include a specific citation for an article or a quick factual lookup such as the name of a capital city or stock quote.
2. The *existence search*, where it is not known if the information exists, or if it does, how it might be described. The difficulty of looking up a feature in software help is illustrative of this type of search; you might know the software can do a needed task, but not know how it is described or categorized in the program’s documentation.
3. The *exploratory search*, where a topic is not well known or understood, making term selection problematic (the dilemma of “how can you look it up if you don’t know how to spell it?”).
4. The *comprehensive search*, where a topic might be well-known, but all possible information on it is desired. For instance, literature reviews on a potential research area where all information on a topic is sought.

These different types of searching frame the examination of Internet search, which is the focus of the next chapter. Internet search is the application of IR principles to the various search engines, metasearch services, and directory subject guides that people use to seek Internet-based information. Each of these various services can accommodate these four general types of search. A URL published in the newspaper or given in a company brochure is a type of known item search. Web directory services or subject guides facilitate the initial steps of an exploratory search. Both existence and comprehensive searches are more challenging and can be problematic for any IR system because it is difficult to prove that something does not exist or that all information on a topic has been located. A Web search engine is the usual starting point for these

searches, and a metasearch engine that broadcasts the search to multiple search engines is a particularly useful strategy to enhance the comprehensiveness of a search. Although the Web was developed to facilitate browsing, the scale of the Web makes simple browsing ineffective unless the search is initially focused by one of the techniques described previously.

SUMMARY

Search has become a defining technology of the Internet. Search engines utilize the principles of general textual IR systems as described in this overview, which examined performance measures, document processing, and the retrieval models that are relevant to understanding Internet search engines. In addition, the Internet is driving many exciting developments in nontextual, multimedia IR. Internet search engines represent IR systems capable of accommodating the various types of searching performed by people seeking Internet information, and this IR overview frames the discussion of search engine technologies in Chapter 15.

REFERENCES

- Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. New York: ACM Press.
- Baeza-Yates, R., & Ribeiro-Neto, B. (2011). *Modern information retrieval: The concepts and technology behind search*. New York: Addison Wesley.
- Battelle, J. (2005). *The search*. New York: Penguin Group.
- Bawden, D. (2007). Information seeking and retrieval. *ALISE*, 28(2), 126.
- Baxter, G., & Anderson, D. (1996). Image indexing and retrieval: Some problems and proposed solutions. *Internet Research*, 6(4).
- Berinstein, P. (1999). Do you see what I see? Image indexing for the rest of us. *Online*, 23(2), 85–88.
- Black, P. E. (2006). Dictionary of algorithms and data structures [online]. Retrieved September 13, 2008, from <http://www.nist.gov/dads/HTML/invertedFileIndex.html>.
- Black, P. E. (2008). Dictionary of algorithms and data structures [online]. Retrieved September 13, 2008, from <http://www.nist.gov/dads/HTML/invertedIndex.html>.
- Brin, S., & Page, L. (1998). *The anatomy of a large-scale search hypertextual Web search engine*. Paper presented at the Proceedings of Seventh World Wide Web Conference, Brisbane, Australia.
- Case, D.O. (2007). *Looking for information: A survey of research on information seeking, needs, and behavior* (2nd ed.). New York: Academic Press.
- Castelli, V. (Ed.). (2002). *Encyclopedia of library and information science* (vol. 71). New York: Marcel Dekker, Inc.
- Cheng, H.-L., & Rasmussen, E. M. (1999). Intellectual access to images. *Library Trends*, 48(2), 291–302.
- Chu, Heting. (2003). *Information representation and retrieval in the digital age*. Medford, NJ: Information Today.
- Chu, Heting. (2010). *Information representation and retrieval in the digital age* (2nd ed.). Medford, NJ: Information Today.

- Dillon, M., Jul, E., Burge, M., & Hickney, C. (1993). *Assessing information on the Internet: Toward providing library services for computer-mediated communication*. Dublin, OH: Online Computer Library Center, Inc.
- Flynn, R. R. (1987). *An introduction to information science*. New York: Marcel Dekker.
- Goodrum, A., & Spink, A. (2001). Image searching on the Excite Web search engine. *Information Processing and Management*, 37, 295–311.
- Horaczek, Stan. (2013, May 27). How many photos are uploaded to the Internet every minute? Retrieved September 11, 2013, from <http://www.popphoto.com/news/2013/05/how-many-photos-are-uploaded-to-internet-every-minute>.
- Jacobs, B. (2010, September 24). How Shazam works to identify (nearly) every song you throw at it. Retrieved July 19, 2013, from <http://gizmodo.com/5647458/how-shazam-works-to-identify-nearly-every-song-you-throw-at-it>.
- Korfhage, R. R. (1997). *Information storage and retrieval*. New York: Wiley & Sons.
- Meadow, C. T., Boyce, B. R., & Kraft, D. H. (2000). *Text information retrieval systems* (2nd ed.). San Diego, CA: Academic Press.
- Morville, P. (2005). *Ambient findability*. Sebastopol, CA: O'Reilly Media, Inc.
- Pepus, G. (2007). Smart image and video search. *KM World*, 16(6), 6–9.
- Rorissa, A. (2007). Benchmarking visual information indexing and retrieval systems. *Bulletin of the American Society for Information Science and Technology*, 33(3), 15–17.
- Rosenfield, L., & Morville, P. (1998). *Information architecture for the World Wide Web*. Sebastopol, CA: O'Reilly.
- Salton, G., & McGill, M. J. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.
- Smeaton, A. (2007). TRECVID-video evaluation. *Bulletin of the American Society for Information Science and Technology*, 33(3), 21–23.
- The Text REtrieval Conference. (2007, August 8). Overview. Retrieved May 1, 2008, from <http://trec.nist.gov/overview.html>.
- Wilson, T. D. (2000). Human information behavior. *Informing Science*, 3(2), 49–56.
- Zipf, G. K. (1949). *Human behavior and the principle of least effort; an introduction to human ecology*. Cambridge, MA: Addison-Wesley.

ADDITIONAL READING

- Chan, L. M. (2007). *Cataloging and classification: An introduction* (3rd ed.). Lanham, MD: The Scarecrow Press, Inc.
- Chowdhury, G. C. (1999). *Introduction to modern information retrieval*. London: Library Association Publishing.
- Lancaster, F. W. (1979). *Information retrieval systems: Characteristics, testing, and evaluation*. New York: John Wiley.
- Yang, K. (2005). Information retrieval on the Web. In B. Cronin (Ed.), *Annual review of information science and technology* (vol. 39, pp. 33–80). New York: ASIST.

This page intentionally left blank

Internet Search

Internet search continues to rival email as one of the most popular Internet activities; 73 percent of Americans reportedly used search engines in 2012 (Purcell, Brenner, & Rainie, 2012). Search is not just an individual means to an end, but search patterns are also sociologically significant as a measure of broader collective interests. Google Trends tracks the most popular searches of the day and provides a fascinating snapshot of what is culturally significant at any given time. This phenomenon represents a “database of intentions” (Battelle, 2005) and is a cultural artifact that records shifting societal trends over time, and many search engines incorporate lists of what searches are currently “trending” or popular. While broader trends provide an intriguing view of the totality of individual searches, it is the individual’s search activities and search engines’ technologies, which respond with useful, relevance-ranked results that are the focus of this chapter.

Search engine technologies are applied to IR in many contexts; there are programs for desktop search engines, enterprise search engines for a company intranet, and topic-specific, vertical search engines. Internet search as described here is IR applied to the specific context of using free search engines to find Web pages, documents, or other forms of content that are relevant to a query on the open Web, whether it be for quick information seeking or to support in-depth research. The focus is on the general process employed by search services to create a searchable index, process a query, and rank the results. The theoretical and mathematical underpinnings of these systems are considered only to the extent needed to support a general understanding of the way search engines work and the consequent implications for searchers. A good text on search engines and IR is from Croft, Metzler, and Strohman (2010).

THE CHALLENGES OF INTERNET IR

Internet IR is challenging for both Internet searchers and search engine designers for a number of reasons. Issues include the extremely large volume of information resources available; the myriad of formats and languages in which content resides; the lack of any consistent quality control applied to published content; the presence of inappropriate and misleading information; the problem of the “invisible” Web; and the lack of any overarching organizational scheme or controlled vocabularies. In addition, success in part depends on the intent and expertise of the user; most quick, factual searches are successful, but in-depth research often requires a deeper understanding of both what is available on the Web and how to best formulate a search. For example, Google Scholar is aimed at an academic audience seeking traditional articles, patents, or case law and provides citation data similar to that of the Web of Science database from Thomson Reuters. However, the searcher might be disappointed to discover that resources found by Scholar are often not available without a fee.

The Size and Currency of the Web

One of the first and most significant issues encountered by developers of all Internet search systems is the sheer volume and diversity of resources that are available and the rate at which they are growing. The growth of the Web has contributed to the notion of “big data”; a Scandinavian research center estimates that 90 percent of all the data humans have produced was generated in the last several years (Brandtzag, 2013). IR technologies that work well in a smaller document universe do not necessarily scale up to the Web environment. The numbers of documents that make up the Internet universe are huge and so are the retrieval sets returned by most searches, making relevance ranking both critical and problematic. Accurate estimates of the number of static Web pages are difficult to come by, and many search engines, including Google, have stopped providing data on index size. However, over the history of the Web there have been attempts to take a census of Internet content. Earlier surveys reported the search engine AltaVista indexed 30 million pages in 1996, 90 million in 1997, and 150 million by 1998 (Sullivan, 2005). Other statistical methods yielded an estimated 320 million pages by the end of 1997, which grew to about 800 million by 1999 (Lawrence & Giles, 1998, 1999). By 2005, the estimated size of the Web had ballooned to 11.5 billion pages (Gulli & Signorini, 2005). Two years later Boutell.com, a software and Internet service company, calculated that there were 29.7 billion pages, but they readily acknowledged this extrapolation was based on a number of fuzzy assumptions for server counts, the “average” number of pages at each site, and Yahoo’s index size, making this little more than a good guess (Boutell.com, 2007). That estimate was also close to the 27.6 billion pages reported by de Kunder (2008). Another estimate (Gil, 2008) suggested there were about 71 billion static Web pages at that time; he now estimates there are over 100 billion static pages (Gil, 2013). Although these data demonstrate that accurate and consistent measures of Web size are problematic, it is clear that the Internet content pool is huge, and it continues to grow at a rapid rate.

In addition to Web size, currency of the content is also a concern. Many pages found in a search are woefully out-of-date, devaluing their information. Many URLs referenced in both printed works and as Web hyperlinks represent dead links or invalid sources where the content has disappeared or been relocated without a forwarding link. One study suggested that almost a quarter of the articles published with URL references may fall into this category (Lawrence et al., 2000). Early on, Kahle (1997) highlighted the need for and the value of archiving the potentially transient content of the Internet; there are websites devoted to this purpose. One notable attempt to create such a repository is the Internet Archive project with its “wayback machine,” which permits a search for pages that are no longer active as well as for earlier versions of a current page.

Format Issues

Another challenge is the diversity of content formats available in terms of both the file formats (other than simple HTML) and in the language of the content itself. Barriers associated with file formats are discussed in Chapter 13. Inappropriate content such as pornography is also an issue, and most search engines have the capability to filter such content. However, Internet searches still yield content that is not only nonrelevant but also potentially offensive to some users.

For a time, OCLC (the Online Computer Library Center) presented a “snapshot” of the Web for a given year. In the last of these studies about a decade ago, OCLC estimated that there were about 9 million discrete sites each with potentially many pages and that about 35 percent of these were public sites that did not require any authorization or fees (Online Computer Library Center, 2002). They reported then that about 3 percent of the Web were “adult” sites. A more recent estimate of the amount of Web pornography suggested it represented as much as 37 percent of the Web, but this comes from a company offering filtering software and could be biased (More than one third, 2010). Other estimates put the figure at a much lower 4 percent, which is consistent with the estimate of a decade ago (Ruvolo, 2011; Ward, 2013). Regardless of the actual proportion of total Web content, the ease of access to pornography and other inappropriate materials continues to be a concern for parents and providers of Internet access to the public.

Language also presents an issue for IR on the Web. Recent snapshots of content on the Web found that about 55.4 percent of the Web was in English, 6.4 percent in Russian, 5.4 percent in German, 4.1 percent in Chinese, 4.4 percent in Spanish, and 3.9 percent in French (Internet world users, 2011; W3Techs, 2013). Some search engines attempt to address language issues with translation services, and a French company has created a multilingual metasearch engine called Qwant (www.qwant.com) that aggregates content from news and Web sources with searches in 15 different languages (Valenza, 2013). The rise in Chinese content over the last decade is not surprising given its concurrent economic growth. While these numbers have shifted over time, these data reflect the concern that the “global” Internet is still dominated by Western language content and, more specifically, English language content. The potential for cultural and language bias was a source of controversy for the Google book

digitization initiative, with some outside the United States asserting non-English works were being marginalized in this effort (Jeanneney, 2006).

Quality of Information

Separate from the potential retrieval of offensive or pornographic content, there is a wide range in the overall quality of the information resources retrieved with Internet search. Evaluation of Web content is discussed in Chapter 13 and it is an IR issue because retrieval sets are often cluttered with items that initially appear to be relevant, but are not acceptable sources due to their low quality, intended audience level, or the presence of deliberate misinformation.

Search engine technologies are susceptible to manipulative techniques designed to influence relevance-ranking algorithms. These attempts have a back-and-forth cycle; as search engines respond, those seeking an advantage in the retrieval-ranking game develop new techniques. For instance, *term stuffing* was somewhat common in the early Web until search engines developed technologies to recognize and combat it. Term stuffing is the repeated placement of keyword terms in a document in a way that is not visible to the user. It might involve placing terms in the HEAD area that are not displayed by the browser or the use of identical font and background colors to fill the “white space” of a document with terms that are present but not visible; either approach could influence an algorithm that correlates term frequency with document relevance. In addition, it is possible to add misleading meta-tag content. Google’s Page Rank has been manipulated through the development of “link bombs,” which are the result of a coordinated effort to manipulate document linking to bring a targeted page to the top of a retrieval set. This issue, along with Google’s ongoing responses to it, is discussed later in this chapter. However, some techniques to improve the findability of a site are not abusive but instead are considered the legitimate application of SEO strategies.

The Invisible Web

The “invisible” Web is content not accessed by most search engines and it represents a huge pool of largely unavailable materials. There are many ways content can become “invisible” to a search engine. The content could be protected behind a firewall, reside on a server that has employed the robot exclusion protocol, be in a Web page with a META tag that excludes the page from being indexed, use a file format that cannot be directly indexed, or be derived from a database. Dynamic database-driven pages do not really exist until a query creates them and unless a copy of the page has been stored in a server cache there is no preexisting autonomous file for the search engine to harvest. As database-driven approaches become more common, Web crawlers potentially miss an ever-increasing pool of this type of content. There are estimates that the invisible Web could contain 400–500 times the content that is available on the indexable Web (Bergman, 2001). Zillman (2007) estimates that the total invisible Web from all sources that cannot be found by traditional Web

crawling search engines could represent about 900 billion pages. It is likely that total is much higher now; Gil (2013) estimates that about 450 billion pages are invisible from database-driven sites alone. Although the boundaries between the visible and invisible Web are shifting and estimates about its size are unreliable, there is no doubt that significant amounts of content are not found by standard search engine approaches. The lack of invisible Web resources in most search engine result sets reduces the comprehensiveness of Internet search as it exists today. Several sources on the importance of invisible Web content and how to find it are by Scheeren (2012) and Devine et al. (2009).

Web Organization

In addition to these problems, the general lack of controlled vocabularies, structured metadata, and organizational schemes has limited Internet search effectiveness. Unlike many formal information systems, there is little organization to the open Web, and keyword searching is the primary access strategy. There have been attempts to create Internet directories using formal organizational approaches, such as the Librarians' Internet Index, Drexel's Internet Public Library, Library and Archives Canada, and the BUBL LINK Catalogue of Selected Internet Resources in the United Kingdom. Such efforts attempt to provide more structured access to Internet content for academic subject access.

There is an emerging interest in creating a metadata-enhanced Web to improve intellectual access to Internet resources. Metadata initiatives, such as OCLC's DC, which is a structured framework for those wishing to utilize metadata, and the broad goals of the "semantic" XML-based Web promoted by Tim Berners-Lee et al. (2001) represent two such approaches. There are also ways to enrich content with microdata or RDFa attributes, as discussed in Chapter 12 and later in this chapter. Although these offer exciting opportunities for enhancing information access, their use is still the exception rather than the rule, in part because the creation and use of structured metadata are labor intensive. There is also some evidence that metadata strategies do not always necessarily improve search accuracy. One study examined queries of a university website that utilized subject metadata and found that metadata-enabled searching actually underperformed when compared to using anchor text, which resulted in better answers to queries (Hawking & Zobel, 2007). However, metadata can also be extracted from raw content and link relationships, and it is being used successfully by search engines with algorithms that support semantic search.

A SHORT HISTORY OF INTERNET SEARCH

From the beginning of the Internet in 1969 until about 1990, the Internet had comparatively little content of interest to the public and there was little need for Internet search as used today. Most early Internet activities were limited to email communication, telnet access to remote hosts, or the use of FTP to access file repositories. These FTP sites could contain large numbers of files,

and locating a specific file of interest was difficult. One of the first Internet search technologies, called Archie, was developed in 1990 at McGill University and was designed to facilitate search of these repositories. Archie was short for “archives” and consisted of an index created by a program that harvested information from the various anonymous FTP sites.

The gopher protocol, developed at the University of Minnesota in 1991, utilized a hierarchically structured menu system to access Internet content. This service grew rapidly, and by 1994, there were about 10 million items on about 5,500 gopher servers on the Internet, making simple browsing ineffective as a search strategy. The VERONICA indexing service was an index of all the terms used in gopher menus and document titles, making direct access to gopher documents possible. VERONICA was supposedly an acronym for the “very easy, rodent oriented net-wide index to computer archives,” but was also named to follow the development of Archie with another of that comic book’s characters (Vidmar & Anderson, 2002). The indexing was not of the full text of the document, so how well the index term represented the actual intellectual content depended in large measure on how descriptive the menu or title words were. The VERONICA service had several mirror server sites, but as demand grew, the service was frequently overwhelmed. Jughead was a similar Gopher search service developed in 1993 at the University of Utah; its name was also drawn from the Archie comics. This service was not very different from VERONICA and some speculate its development was in part simply to complete the trio of search services named after the comic series characters (Vidmar & Anderson, 2002). These early services allowed for Boolean searching and word stemming with truncation symbols, but advanced searching required a complex set of search option modifiers represented with a dash and a letter. The command-based search interface was not particularly intuitive compared to search engines of today.

Also in 1990, Brewster Kahle of Thinking Machines Corporation (who went on to codevelop the Internet Archives project) along with other corporate partners developed the Wide Area Information Server (WAIS). The software could create a database of weighted terms that was searchable with natural language and yielded a relevance-ranked output. This technology was an early precursor to the modern search engine.

Tim Berners-Lee developed the World Wide Web during this same period. In late 1991, he publicly demonstrated the Web for the first time at the Hypertext ’91 Conference in San Antonio, Texas. The original WWW ran on a server at CERN, which is where it was developed, and access to it was through text-based browsers running on UNIX hosts. In 1993, NCSA released the Mosaic graphical browser, and by the next year, there was an explosion of Internet traffic, increasing an astounding 1,500 percent (Cheong, 1996). The saying “if you build it they will come,” which is often applied to the Internet story, apparently should be restated as “if you build it and *make it really easy to use and useful to people’s lives*, they will come.”

Initially, you could browse the embryonic Web simply by logging into the info.cern.ch site with a telnet connection. The ease of browsing is also apparent in the simplicity of the first browser view of the CERN site, re-released in celebration of the 25th anniversary of the Web (<http://info.cern.ch/hypertext/WWW/TheProject.html>). However, as the Web grew in scale, simple browsing

was no longer a viable strategy without the initial ability to identify promising starting points for browsing. In 1993, the ALIWEB (Archie-like index of the Web), one of the first Web crawlers, was developed. ALIWEB harvested and combined local index files created by a webmaster into a large-scale searchable index. Oliver McBryan developed the WWWW (also called the W4 worm) in 1994, and it expanded automatic indexing to include Web page title words and URLs (Vidmar & Anderson, 2002). Advances in Repository-Based Software Engineering (RBSE) resulted in two companion “spider” programs; one that would find URLs and another that would retrieve and index the documents associated with those URLs. The robot programs that automate the process of creating a searchable index for Web resources are the foundation technologies of modern Web search engines.

The development of these crawler programs in 1994 led to the rapid development of various search engines that continues to the present. The directory services EINet Galaxy and Yahoo, as well as the WebCrawler, Lycos, and Excite search engines, were all developed that year, followed in 1995 by Infoseek, AltaVista, and Inktomi. Northern Light appeared in 1997 and introduced the innovation of categorizing results into folders. Google, released in 1998, quickly became the most popular of the search engines. The metasearch engine, which allows a query to be processed by multiple search engines simultaneously, appeared in 1995 with SavvySearch and MetaCrawler, soon joined by Ask Jeeves and Dogpile, among others. Since that time, hundreds of search engines have come and gone, and there are many successful niche search engines. Google, Yahoo, and Bing share most of the search marketplace today.

INTERNET SEARCH SERVICES

Internet search services consist of three fundamental types: the Web directory, the search engine, and the metasearch engine. Early in the development of these services, the lines between them were sharply drawn, but now there is much functional overlap; directory services also have a search option, and many search engines support directory-like categories of content.

Directories

Directory approaches involve the creation of a categorized list of topics, usually with hierarchically structured menus. Resource selection and categorization may take place automatically or with human intervention. The human intervention may be by employees of the service or by the Web community at-large. Directory services can be especially effective when someone wishes to explore or expand a known topic. Directories provide useful starting points for an exploratory search by identifying the various facets of the topic. One of the difficulties with directories is that the topic headings chosen by their creator may not reflect categories or terms that match how the user thinks of the topic; the issues of polysemy and synonymy discussed in Chapter 14 are relevant here. One solution is to standardize the categorization scheme and to make a subject thesaurus available. A directory service must also define the comprehensiveness with

which it attempts to represent all possible topics as well as define the intended target audience. Directories for general audiences may have top-level categories for sports or entertainment as opposed to academic topics, which limits their effectiveness for more serious research. Hierarchical directory menus can also be problematic because a user must decide on the best entry point to the directory tree structure and then drill down far enough to determine if that initial choice was correct. To make directory use more efficient, a search feature can assist the user in identifying the most appropriate entry point rather than requiring the exploration of many menu layers to find an entry.

One example of a directory service is the one created by Yahoo. Yahoo began its business with its directory service but has since evolved into a complete Web portal offering a variety of other services such as mail, Web search, Yahoo Answers, and Yahoo messenger. Yahoo built its reputation on the claim that although it did not have the “biggest” index; it provided access to the “best” Web content on a given topic because of its resource selections. The DMOZ Open Directory project is an example of a directory that uses the community of the Web itself to create it. This collaborative approach allows individuals interested in a topic area to contribute selections to it. Figure 15.1 is a screen shot of the DMOZ Open Directory categories.

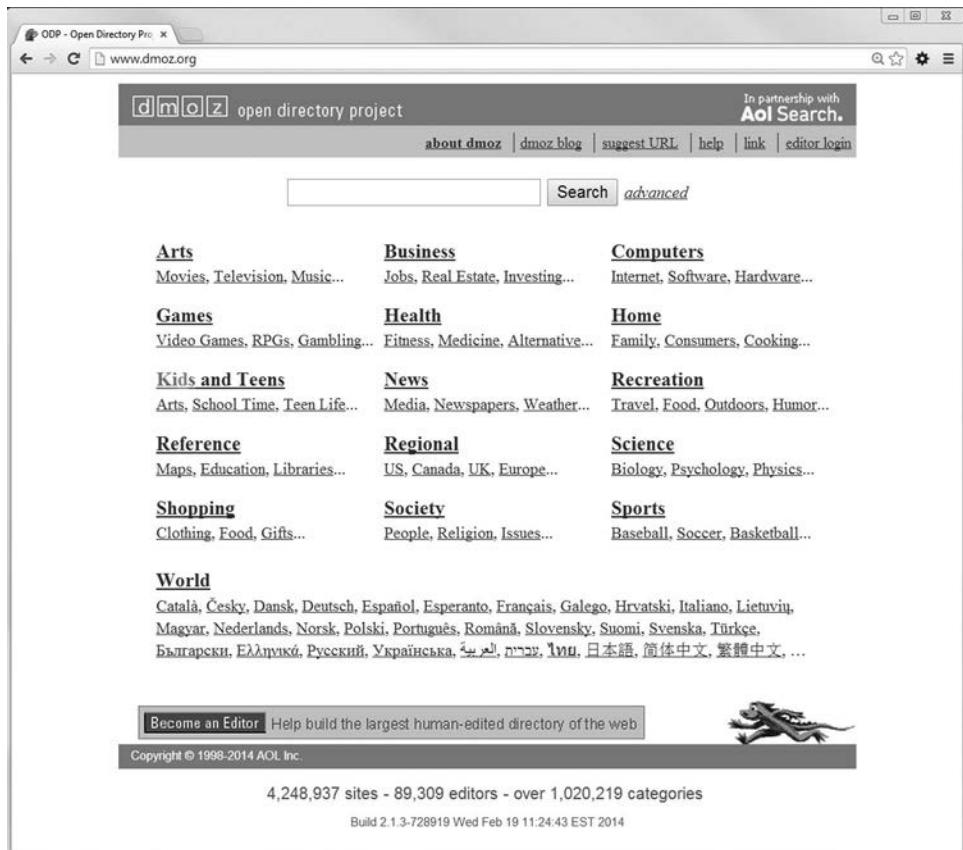


Figure 15.1 The DMOZ open directory project (<http://creativecommons.org/licenses/by/3.0/>).

Search Engines

From 1994 on, the number of search engines grew rapidly and there are hundreds of Internet search engines to choose from today. Although they offer different features, most use similar technologies to automatically create a searchable index and process submitted queries. They all seek to rank results by relevance, but the details of that process are often proprietary algorithms that distinguish one service from another. Search engines facilitate the exploration of a topic with simple keyword searching or through smart search technologies, attempt to answer a natural language question. They are a useful starting point for both a known item type search and for an existence search, where the user needs to determine if information on a topic is even available. The emergence of algorithms that support semantic search means search engines are not just a way to find links to content that may have an answer, but they can provide direct answers to anticipated questions within the results page itself.

Search engines vary considerably in index size and focus; some attempt to index as much of the Web as possible while others focus on a smaller, more specialized niche. There is also much variation in the type of materials indexed; all index HTML content, but many also index content in other formats such as PDF. As with directory services, many search engines have expanded to become complete service portals offering services such as email, maps and driving directions, satellite imaging, image search, language translation, and access to other types of digital content such as that provided by Google Scholar and Google Books. There is considerable variation in both the user interface and in the search features offered among search engines. Many favor the simple interface design that has been so successful for Google, relying on a single text box for users to enter keywords; others provide a more complex set of options on the initial search screen. Even the simple text box approach has potential pitfalls depending on how well the user understands the assumptions made by the search engine designers. A multiple word search has a number of implicit assumptions, and to understand the outcome of a query the searcher must be aware of them. For instance, what happens when more than one word is entered in the query box? Is the default to process these multiple word searches as a Boolean AND (all the words) or a Boolean OR (any of the words)? Does word order make a difference? Does the use of capital letters in the search terms affect how the search is processed? Does the search engine see the query as a string of various keywords or does it treat them more holistically as a semantically meaningful natural language question? The answers to these questions matter, and various search engines handle them differently.

Many search engines have common approaches to advanced search features such as the use of quotation marks to indicate a phrase search, or the use of a plus (+) or minus (-) sign to force a term to be present or absent. For instance, a searcher interested in World War I zeppelins but not the rock group Led Zeppelin might be able to exclude potential false hits by the search “zeppelin -Led.” It is also usually possible to restrict the search term to occurrences in a specific field by naming the field and separating it from the term with a colon. For example, a Google search for the term “diet” could be searched as “intitle: diet” to restrict the search to the HTML title field or “inurl: diet” to search for the term only within the URL. The use of “link:” followed by a URL of interest is a way to

identify pages that have linked to that URL. The domain can also be restricted to a specific TLD, for example, “site:.gov” retrieves results from pages housed only on government sites. Wildcards are often permitted; for example, Google uses the asterisk (*) to substitute for any number of characters and the tilde (~) can precede term to include synonyms in the search.

Although advanced search options are useful, some of these features are becoming less necessary as search engines have become better able to disambiguate the possible meanings in a query through semantic techniques or within the results page. For example, the word “zeppelin” entered in Google by itself now results in a knowledge graph with options for both the airships and the rock band. Many search engines allow for complex Boolean searching as well as restriction techniques such as limiting results to date ranges, media type, or Internet domain location, but they also can employ “fuzzy Boolean” logic to the words in the query to provide relevant query phrase suggestions. All these features require some exploration of the advanced options that ideally are well documented on the site. A good overview of practical search tips and techniques for using various search engines is by Hock (2013).

Metasearch Engines

Metasearch engines enable simultaneous search processing by many individual search engines. This approach is very useful for existence searches, and it is especially helpful for comprehensive searches, where as much information as possible is sought on a topic. There are large differences in the coverage of the major search engine indexes and as will be discussed later, there is sometimes little overlap among them. Consequently, the total reliance on just a single search engine can result in the omission of potentially important retrievals. The metasearch engine attempts to solve this problem by broadcasting the query to multiple search engines or by creating an index that combines multiple indexes. The metasearch engine removes duplicate retrievals from the resulting set and typically sorts the output by the search engine it came from or with some type of dynamic categorization.

There are disadvantages to the metasearch approach. First, the search results are dependent on the underlying search engines the metasearch happens to include, and sometimes that array of search engines is not optimal for a particular search. Second, not all search engines support the same feature set, and various “power searching” options are often not available. Even when a metasearch engine allows inclusion of advanced search options, the search results may still be different from what would be obtained directly from the individual search engine itself. Popular metasearch engines include Metacrawler.com, Dogpile.com, Surf wax.com, and Mamma.com, to name just a few.

THE ECONOMICS OF SEARCH ENGINES

On the surface, the economics of search engines seems somewhat puzzling as they apparently provide services with no cost to the user. Before Web search engines, online services such as Dialog and Lexis-Nexis developed a business

model that required user accounts to be charged for searching based on the amount of time connected, the number of searches performed, the number of records viewed or printed, or some combination of these approaches. There were a few early attempts to mimic this fee-based model with Web search, but the overall Internet culture of “free access” worked against it. For instance, InfoSeek initially began with a pay as you go model but abandoned it by 1996. There are also opposite precedents—the Northern Light search engine began as a free service but switched to a fee model that found later success in the corporate marketplace. The economic model of search engines is similar to that of commercial television—it is technically free, but a number of minutes for each program slot are devoted to advertising that viewers may or may not want to watch.

Search engines are technology companies, but they are also media companies because most of their revenue comes from advertising (Mills, 2005). Some of these companies have been quite successful financially; best known among these is the juggernaut Google. Google’s reported revenues of \$6.139 billion in 2005 grew to nearly \$50 billion in 2012. This growth was mirrored in its stock share price, which has grown from about \$200 per share to a high of about \$900 per share during that period. Others, such as Yahoo are holding their own and some, such as Bing, appear to be gaining market share (McGee, 2013).

Advertising income is generated in a variety of ways. It comes from paid banner ads, which all users see, the “click through” model, where payment is based not just on ad placement but on the number of users who click through to the site, and the pay-for-action approach where advertisers pay when the ad leads to a specific action such as a sale (Broder, Fontoura, Josifovski, & Riedel, 2007). There are also sponsored links that result from “term leasing,” where advertisers bid on associating their link with a specific search term. Sponsored link ads targeted to the search query may be useful for certain types of searches, but they can also be distracting. Most search engines distinguish sponsored links from other retrievals by visually segregating them from the rest of the retrieval set. There are concerns that payment by advertisers could influence the ranking process in ways that would not be apparent to the user. A related issue is the potential for search engines to track user searching and sell or report this information to a third party. Privacy concerns also arise with personalized searching based on a user-supplied profile. Google uses several of these techniques to generate income for both itself and sometimes for site owners as well. Google’s AdSense allows site owners to place code in a location on their pages, receive bids from potential advertisers, and then receive payments from them through Google. Google’s AdWords allows you to associate search terms with ads displayed in Google, which result in charges only when users click through and land on your site.

Search engines have other potential sources of income such as payments from ISPs or corporations for inclusion of a toolbar or link to the search engine, or revenue from marketing the search technology as a site search appliance. Nevertheless, the dominant economic activity that supports most of these companies is income from various types of advertising. There have been many buyouts and mergers of these services and some “outsourcing” of certain functions to another engine or directory.

INDEX SIZE AND INDEXICAL BIAS

Over the early years of the Web, search engines engaged in promoting their index size both in relation to other search indexes and to the estimated size of the entire Web. Sites such as Search Engine Watch and Search Engine Showdown compared index sizes of various search engines over time, but many services no longer report these data. However, it is reasonable to expect that some of the trends from previous index size studies indicate there could be potentially large differences in the comprehensiveness of these indexes. One such early comparison done by Notess (2002) found Google was a clear early leader with about 3 billion pages indexed, which was about twice the size of AltaVista and about three times the size MSN Search. As of 2013, Gil (2013) estimated that Google indexed 40 billion public Web pages, an impressive accomplishment to be sure, but that represents only about 40 percent of the estimated total number of Web pages. Even as that index continues to grow exponentially, there is always the question of how comprehensive it is at any given time.

Studies comparing results of a standard query sent to various search engines documented significant differences in index coverage over time. Notess (2002) raised this issue early on by analyzing the results of four simple, standardized searches on 10 different search engines. He found 334 total hits, 141 of which were specific Web pages. Of that set, about half (71) were found by only one of the 10 search engines; another 30 of these unique hits were found by just two. A 2007 research report from the metasearch engine Dogpile that compared large sets of user-defined queries across Google, Yahoo, and AskJeeves found that only 1.1 percent of the first pages results were the same across these search engines concluding that the results provided by different search engines were relatively unique (Dogpile.com, 2007). Index overlap is also described as *indexical bias*. The word *bias* can have a negative connotation, but in this context, bias refers simply to the hidden assumptions that are inherent in the design and implementation of search engines. Mowshowitz and Kawaguchi (2002) examined indexical bias differences among search engine indexes, finding that the identical query of “home refrigerators” resulted in considerable differences among the top 50 retrievals gathered from nine well-known search engines. These studies suggested that indexical bias is of theoretical interest and a practical concern and that comprehensive searching could benefit from the use of more than a single search engine or a metasearch engine to compensate for inconsistent and variable index coverage.

Although index differences among search engines are still a potential issue, these studies of indexical bias predate recent advances in search technologies. Personalization, location-based searching, and semantic search, along with specific adjustments to ranking algorithms designed to combat result spamming, can all also contribute to significant differences in search engine result pages (SERP). With search personalization and semantic search, there is no longer a typical “first” page of results; a given search engine might present different results to different users because it interprets the semantic “intent” of the search differently for each. Differences among SERPs for the same search done on different search engines can be attributed to many factors in addition to index differences.

INTERNET SEARCH ENGINE ANATOMY

Search engines and their associated programs perform the essential functions that facilitate Web search. They identify URLs of pages to index, manage the resulting URL database, harvest content from identified resources and parse it into an index, provide the search-user interface, accept queries and match them to the index, and present ranked output to the user. Along the way, these activities perform other useful secondary functions, such as providing estimates of the size of the indexable Web, identifying dead links, links that point back to a specific page, and caching pages to create an archive of pages no longer available. What follows is a general description of this process.

The URL Database

The first step to building a search engine is to identify Web URLs. Most search engines accept URLs submitted by content producers, but the vast majority of the URL database is the result of the actions of spider programs. The process of automated discovery begins with a Web crawler program, also known as a spider or more generically, a robot, retrieving all URLs it can locate. Set loose on the Web, these programs follow every link they find according to program parameters that determine how they are to proceed. Spider programs may follow Web links using a breadth-first or depth-first strategy. Breadth-first, the most common method, is when the crawler follows the first link in a page and then all the links in that secondary page before returning to the first page to repeat the process with the next link it contains. In the depth-first approach, the first link in each page is followed, so the first link in a second-level page would be followed, and then the first link in that page, and so on with the result that the spider drills down as far as it can before returning to the original page with that first link. Both approaches require a queue of URLs to be tracked; the breadth-first approach adds new links by appending them to the end of the queue in a “first in first out” strategy; depth-first spiders use a “last in first out” strategy that adds new URLs to the front of the queue. Spiders differ in how deep they are programmed to go into a site and in whether they follow all links in a page or just a sampling of them. These programming directives to their spiders account for the differences in size and coverage among search engine indexes. The URL database is then sorted and de-duped, resulting in the grouping together of all items from the same domain. Many Internet resources are not included in this process. There are many ways content may be overlooked by a Web crawler, including:

- Non-HTML formats such as PDF files as well as other formats are often not indexed. However, the formats indexed by search engines vary; Google started indexing PDF in February 2001.
- HTML coding can affect spider crawling results. For instance, frame-set files may be independently crawled, resulting in access directly to those pages outside their intended viewing context. In addition,

JavaScript code in the HTML is ignored by indexing programs, so pages that depend on scripts are not fully represented in the index.

- The Robot Metatag with the “noindex” directive within a specific page turns the spider away from that page.
- Images that appear in a page are indexed only by the filename or with alternate text in the image tag, if available, making image search problematic.
- Spiders cannot access sites that require authentication or registration.
- The Robot Exclusion Protocol, a “netiquette” for robots that is respected by most search engines, can turn away the spider. The site administrator can insert a file called robot.txt with restrictions to the visiting robot to keep away from the server or restrict specific areas on it.
- Dynamic Web content that does not yet exist as a static HTML page that can be indexed.

These nonindexed sources by definition become part of the “invisible” Web previously described. However, Web developers may be interested in ensuring a page is included in a search engine index as opposed to keeping it private. One solution is simply to submit the URL; most search engines allow a content creator to submit a site requesting inclusion. The other main ways to increase the odds of inclusion is to increase the number of page in-links or enhance the authority of the page by other means discussed in the section on semantic search; such intentional SEO techniques are considered part of the design process.

Creating the Index

The next step in the creation of the index involves a robot crawler program, sometimes called a harvester. Its job is to visit the pages in the URL database and harvest content for parsing into an index. The amount of content gathered varies; typically, it is the entire page, but it may be just the head area along with a portion of the body content. The collected pages make up the document store to be processed. Stop words are typically (although not always) eliminated as nonmeaningful terms, and the remaining document terms are melded into an index in the form of an inverted file. Each term has a pointer back to the documents the term appeared in as well as other information such as its frequency and position. The result is a record that includes a document identifier, information about the location of a term in a document, the number of occurrences of the term, the total number of words in the entire document, and the HTML tagging associated with it, all of which provide data to the relevance-ranking algorithm.

Simple keyword indexes can connect terms to points of Web content and contribute to some assessments of probable relative relevance to a query. However, HTML link tags and generic metadata elements offer only limited information about the resource, its creator, and how it relates to other sources. Semantic

search requires a semantic index that includes metadata in the form of formal schemes and microdata markup, as well as the ability to utilize the relationships of the data to other sources or particular content creators (Amerland, 2014).

The User Interface

The user interface provides the functionalities needed to present a search to the query processor. The interface is an HTML form, and scripts process the data the user submits and translate it into an acceptable query for the backend server. The conversation that takes place between the client and server with such a form uses a GET or POST method, discussed with HTTP in Chapter 5.

Such a form interface can be quite simple, as with the well-known single text box of the Google search, or more complex with options the user can select. There are many design issues embedded in the creation of this interface that affect its success and use. The best underlying search engine technology will not compensate for a poorly designed, cluttered, and difficult-to-use interface. There is an obvious tension between the competing objectives of providing a clean, easy-to-use interface with the concurrent economic need to place banner ads and sponsored links within the same page. Some design tradeoffs tip the balance one way or the other. Some sites do not place any ads on the initial search screen and clearly segregate them in the results, while other services seem to pack in as many ads and pop-ups as possible. The availability of online help and the ease of finding it is another interface issue that differentiates search engines. Some services provide excellent documentation with many search examples; others are almost totally lacking in such documentation.

Query Processing and Relevance Ranking

Once submitted, a query processor handles the search and then ranks the results. These are separate search engine modules, but taken together, they comprise a key functionality that distinguishes one search engine from another. Query processing involves converting the natural language of the searcher into a query the system can accept, executing it, and returning a retrieval set. The ranking module determines the size of the set presented and the ranking order. Most search engines view their query processing and result-ranking algorithms as proprietary information and do not make all the details of this process available.

Search engines make use of the classic IR models discussed in the previous chapter, applying them individually or in some combination. Many rely on the Boolean model, but term weighting expands strict Boolean to incorporate factors such as the number of occurrences of the term compared to the size of the document and the order of the terms in the query. A simple Boolean search of the inverted file usually determines an initial retrieval set; this set might be returned to the searcher as is or additional processing using term weights could be performed. Some search engines have incorporated elements of the vector

space approach. Early on, the Inktomi search engine utilized the vector space model, but the full implementation of this model has been constrained by both computational expense and scalability issues (Langville, 2005). Search engine use of the probabilistic model is also limited by its complexity and scalability. However, some probability-related approaches have been successful; the creators of the Google PageRank algorithm describe it as an attempt to determine a probability distribution derived from link analysis. In addition, Google uses fuzzy Boolean logic to offer search phrase suggestions based on a semantic analysis of similar past queries that started with the same keyword.

Link Analysis and Retrieval Rankings

Link analysis can have multiple connotations depending on the kind of “link” that is examined. The term is used in the context of national security intelligence gathering and data mining, where it applies to following communication links to determine a social network of connected people. In the context of Internet search, however, it refers to algorithmic ways of exploiting the hypertext structure of the Web to influence the filtering and ranking of a retrieval set as well as to derive semantic relationships among terms. Conceptually, link analysis is similar to citation analysis in the premise that a measure of the importance of a work within a discipline is associated with both its citation frequency and the prestige of the sources that cite it; it is reasonable to assume that a work cited many times is important in that discipline. Citation analysis provides a metric for measuring the impact of journals known as the *journal impact factor* (Garfield, 1972). In the hypertext Web, embedded links are a form of citation, so it is not surprising that link analysis could contribute to both creating retrieval sets and their subsequent ranking. However, link analysis techniques, while tied to the idea of citation analysis, do not depend exclusively on simple numbers of links to determine a measure similar to Garfield’s impact factor. There are notable differences between citation patterns within printed scientific literature and the nature of Web hyperlinks that must be considered in search engines implementations (Chakrabarti et al., 1999). The goal of a search engine is to deliver Web pages that are judged both *relevant* and *authoritative*. Although this judgment is ultimately a human one, link analysis and metadata techniques can provide viable algorithmic solutions.

Although search has evolved significantly beyond simple link analysis, it is still an important part of how Web search engines automate result rankings. Two models for hypertext link analysis have been used in Web search; perhaps best known is PageRank developed by Sergey Brin and Lawrence Page, the Google founders. However, Brin and Page were not the only ones working on link analysis during the 1990s. During the same general period they were developing Google at Stanford, Jon Kleinberg had developed the Hypertext Induced Topic Search (HITS) model (Kleinberg, 1999). Although Kleinberg did not develop this concept into a public company, as did the Google developers, a prototype engine called Clever was developed (Chakrabarti et al., 1999). In addition, the Teoma search engine also adopted this algorithm (Langville & Meyer, 2006). Both the HITS and PageRank models are of interest to this discussion.

Relevance is not easy to determine automatically, and the nature of the query plays a role in how difficult the task is for a program. Very specific queries are easier to handle as the retrieval set tends to be relatively small, and the judgment of relevance is based on the presence or absence of the “answer” sought. However, general queries usually retrieve larger sets, and the degree of fit to the query is harder to assess automatically. Kleinberg (1999) observed that specific queries are affected by the *scarcity problem*, that is, there may be few pages relevant to a very specific query and it may be difficult to find them, and broad queries give rise to the *abundance problem*, that is, the set of pages returned can be too large for a person to digest. Further, he notes that within a very large retrieval set, a process that would filter the set to retain the most *authoritative* sources is desirable, which is the primary goal of the HITS model he developed. Assessing authority is the subjective process of evaluating the significance and trustworthiness of the information presented. Kleinberg’s (1999) work suggests that link analysis can provide an algorithmic solution to this problem. The key idea of HITS is that if Page A links to Page B, then the creator of Page A must have deemed Page B to be worthy of the link; in that way Page A has *conferred* some degree of *authority* to Page B. The ideas of authority and trust have been broadened in the context of semantic search discussed later in this chapter.

In the HITS model, pages are *nodes*. The model presumes that a query has yielded some initially large retrieval set, and that set is further restricted to a relatively small subset consisting of the highest ranked retrievals. This subset is referred to as the *root set*, and it can be expanded to form a base set by including other pages identified by having links to and from any of the root set pages. This expanded set can then be further refined through link analysis with the goal of finding pages that are not just relevant but also the most *authoritative*. The HITS model attempts to identify these computationally. Within the root set, there are pages that link to other pages within that set. The number of pages that link to a given page provides a measure of the *in-degree* for that page, and the number of pages it links to is the *out-degree* of that page. Pages with a high measure of out-degree are *hubs*, and pages with high in-degree are *authorities*.

In this model, good hubs point to good authorities, and good authorities are linked to good hubs. This sounds circular, as does the PageRank process discussed later, but Kleinberg’s (1999) work showed that if this iterative algorithm is run a sufficient number of times, the results converge to values that turn out to be useful in determining relevance. In his model, the identification of hubs and authorities is therefore mutually reinforcing, and hubs can serve to connect together different authority groupings within a specific subject area.

GOOGLE: AN EXAMPLE OF A LARGE-SCALE SEARCH ENGINE

Google has clearly grown beyond search to become a multifaceted technology company. However, its success was built on its search engine and it continues to be a leader in that market; between 2005 and 2012, its market share grew from 45 percent to 67 percent of all Internet searching (Adamo,

2013; Goodwin, 2013; Mills, 2005). Although both Yahoo and Bing have made slight gains in 2012 with 12.2 and 16.2 percent of all searches, together they still are less than half of Google's share (Goodwin, 2013). Google is not just the name of a company, but a term used as a verb (to "google" something) as well as a trend ("googlization"). Since its inception as a search engine, it has grown to offer a huge array of other services including YouTube, Gmail, Google+ and Google Hangouts, as well as numerous cloud applications and services. The company also competes in the hardware arena partnering to build the popular Chromebook computer and developing the next generation of wearable technology tools such as Google glass. Google has also become an important service provider for email, cloud storage, social networking, and other Web 2.0 services.

Google search warrants a closer look in a discussion of Internet search because it is a recognized leader in the search marketplace, and as such, it frames the expectations of the public regarding interface, ease of use, and search efficiency. In addition, Google was the first large-scale search engine to use some novel approaches to relevance ranking, including PageRank, an interesting application of link analysis. A detailed explanation of the mathematics of PageRank is by Langville and Meyer (2006).

Google History and Background

The success story that is Google is well known. Developed in 1998 by Sergey Brin and Larry Page while they were graduate students at Stanford (Brin & Page, 1998), Google's search architecture utilizes a link-based retrieval model they had been working on since 1995 called PageRank (PR). The company name was derived from the word "googol" or 10^{100} (10 raised to the 100th power, a huge number (Brin & Page, 1998).

Google has become a Web phenomenon, handling about 543 million searches per day by 2007 (Smalera, 2007). As of 2013, this has grown to over 3.3 billion searches per day; interestingly, nearly 500 million of those queries are new to the search engine (Bort, 2013). Google has always strived to be the largest, most comprehensive of the Web indexes, and its crawlers reportedly have identified about 30 trillion unique Web addresses (Bort, 2013). The sheer volume of data that it stores is difficult to assess accurately, but as of mid-2007, data from Google itself led some experts to estimate it was managing at least 20 petabytes of data on its servers; some reports speculated this figure could be as high as 200 petabytes at that time (Smalera, 2007). A petabyte is about 1 million gigabytes, or about 1,000 terabytes. Numbers that big are hard to conceptualize, but to put this in perspective, consider that if a 30-gigabyte iPod can hold about 10,000 songs, a 1-petabyte iPod could hold about 300 million.

Google is impressive, but is it "smart"? Artificial intelligence (AI) has represented a "holy grail" of computing throughout its history, from the time of the proposed "Turing test" to the present. Traditional AI research has sought to create a "thinking" program that can interact with people (think of HAL in the film *2001: A Space Odyssey*), but much traditional AI research has shifted focus to the relationship between embodiment and intelligence as needed in the field of robotics (Pfeifer & Bongard, 2007). Google is not an "intelligent" program by

traditional AI standards, but it can make inferences from its data. In a 2003 colloquium on data-driven texture and motion in graphics, Alexei Efros (2003) referred to Google as AI for the postmodern world, observing that whatever “intelligence” Google has is in the data itself, as represented in its index and the queries it receives. In this view, all questions have already been asked and answered; the challenge is not to create an answer, but to find one. Google’s term spelling suggestions are from data associations and not a dictionary. When Google asks, “Did you mean . . . ?” it reflects an inference that certain term combinations or spellings are much more common than others (Efros, 2003). Now, after 20 years of search, Google has both “big data” and new algorithms that further exploit these data relationships to support semantic search. Google no longer needs to ask “Did you mean . . . ?” but saves a step by simply providing the corrected search in the drop down window of search phrase suggestions as you type. Search engines are evolving beyond search and becoming answer engines—not quite “Star Trek” yet, but getting closer all the time as users of Google voice search or Siri will attest. Semantic search and examples of its implementation by Google and others will be explored further in this chapter.

Google Infrastructure

The high level of search demand and the huge amount of data Google manages require novel data management and parallel processing techniques. Google’s data resides on about 200 server clusters in multiple data centers around the world. Google’s sophisticated networking software allows the use of relatively inexpensive PCs that have two dual-core processors, two hard drives, and cooling fans to dissipate heat (Barroso, Dean, & Hözle, 2003). Each rack can hold up to 80 such machines, and parallel processing techniques allow the replacement of drives or whole PCs without interrupting service. Google depends on thousands of commodity class PCs with fault tolerant software instead of fewer high-end servers, a creative strategy that they have found to be a more economical approach to handle its enormous workload (Barroso et al., 2003). Estimates vary, but a 2011 report suggests Google has over 900,000 servers running using about 220 megawatts of power; a new storage and networking system called Spanner is planned to automate data management across a network that could grow to as many as 10 million servers across multiple data centers (Miller, 2011).

Google Architecture

The decision to use commodity-class PCs forced the developers to treat component failures as the norm as opposed to the exception. The resulting Google File System (GFS) economically handles the large-scale workloads but regularly and transparently repairs any damage such failure can cause to data integrity (Ghemawat, Gobioff, & Leung, 2003). Google runs three crawlers simultaneously that can crawl about 100 pages per second. Web documents are cached, assigned a document ID, and stored in compressed form in a repository. Document words, called occurrences or hits, are assigned a word ID to

create a lexicon of over 14 million words. Information is stored about each hit, such as its position in the document as well as text decoration and capitalization. Links are extracted and stored in an anchors file, which allows further link analysis and weighting. This index can connect keywords to resources and yield relevance-ranked results links, but semantic search needs a semantic index. Google's index still has this same Web data, but it is enhanced with new sources of metadata and microdata. It also uses a database of personal profiles or data from other social signals in ranking the trustworthiness and authority of a site. For example, indexed content that is associated with someone with a huge Twitter following on a topic or Web content that is heavily commented on or highly reviewed in social media can be given more authority and thus be ranked higher in a search (Amerland, 2014). Google compresses and stores all this information, as well as data for its many other applications, in a proprietary storage system called BigTable, which is distributed across thousands of servers (Chang et al., 2006). The original Google Web index system presented by Brin and Page (1998) is shown in Figure 15.2.

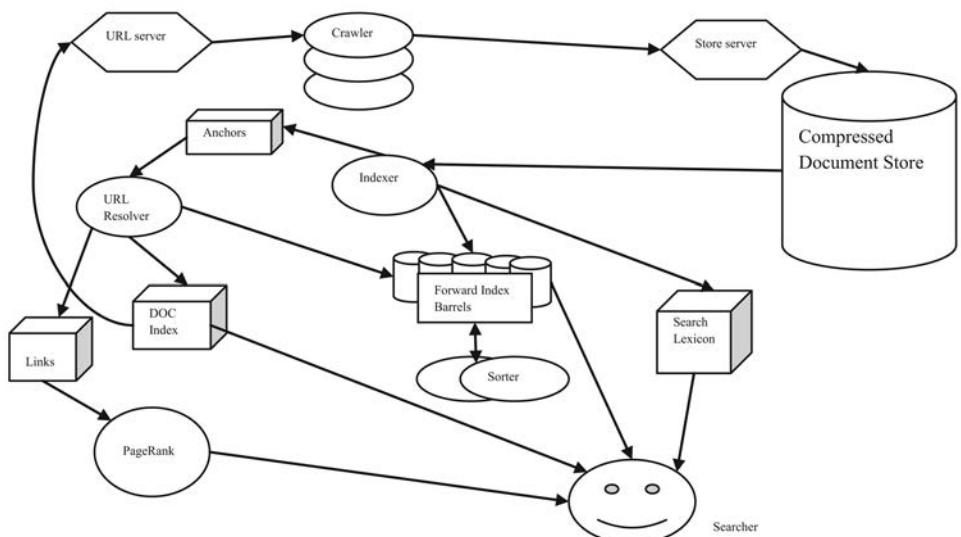


Figure 15.2 High-level view of the Google architecture as diagrammed by Brin and Page (1998). Multiple crawlers gather pages from lists of locations provided by the URL server, which are then sent to the store server and assigned a DocID, and a compressed form is saved in the document store/repository. The indexer uncompresses and parses the pages, and the terms, numbers of occurrences, and locations in the document are stored in “barrels” to form a sorted forward index. The indexer also extracts link information and stores it in the anchors file with the result that anchor text words in the forward index have pointers to the DocID of linked resource. The URL resolver uses the anchor’s file to store information about in-links and out-links to determine PageRanks. The extracted document words also form the lexicon of search terms for the searcher (Brin & Page, 1998).

Google and Relevance Ranking

Google's reputation and success are built largely on the efficiency of its relevance ranking of results. In addition to PageRank, Google makes use of other document features, such as text decoration. For instance, text in bold or in certain HTML fields is weighted more highly to reflect the emphasis given to the term by the document creator. Google also handles anchor text differently than some search engines. Instead of treating anchor text as other words in the document that are taken to represent the page they come *from*, Google treats anchor text as descriptive of the page the link *points to*. The creator of the page has actually already made that determination by choosing those words for the link; anchor text is therefore a form of human-assigned metadata for the linked resource. Another advantage of recognizing anchor text as unique is that links extend the coverage of the index to those sites, even if the Google Web crawler has not visited them. Google is also advancing semantic search technologies that enhance both query formation and the richness of search results. Over time, Google has made many changes to how PageRank is implemented and their algorithm has become much less dependent on it; with the 2013 Hummingbird revision PageRank is mentioned as just one of the hundreds of factors used by Google to rank results. PageRank has evolved to include more than just links, but its core idea is still about estimating the authority of a site. Its use in ranking results is one of Google's best-known early innovations and as such warrants closer examination.

PageRank

On April 1, 2002, Google posted a description of their relevance-ranking technology, described as "PigeonRank," which utilizes low-cost "Pigeon Clusters" (<http://www.google.com/technology/pigeonrank.html>). Of course, this was an "April Fool's Day" joke—Google does have a sense of humor. While this view is entertaining, PageRank does not have anything to do with pigeons. Google's PageRank algorithm utilizes link analysis for relevance ranking by viewing hypertext links as a "vote" for the importance of the pages linked to by the author making the link. Brin and Page (1998) describe PageRank as a representation of a hypothetical view of user behavior, described with the following scenario: Assume a Web user seeks a Web page only by browsing and following links but can never use the browser "Back" button. At any time, the user can decide to request a new random starting point for their browsing. The probability that the user visits a particular page is its PageRank; the probability that they request a new starting point is the damping factor used in the PageRank algorithm. Although PageRank remains a key component of Google's relevance ranking of results, Google reports using over 200 unique signals in determining the final ranking, and they continue to refine these algorithms to improve the quality of their results (Singhal & Cutts, 2011). Some specific changes to the Google algorithm are explored further later in the next section.

A simplified overview of the mathematics of this model is not too daunting and is sufficient for a basic understanding of how PageRank works. First, there are some terms to define. *PageRank* (PR) is the numeric ranking determined

by Google for each page and used as a factor in relevance ranking; higher PR values imply higher relevance. A *backlink* is when a page has a link to it from another page. Brin and Page (1998) describe the algorithm as follows:

We assume page A has pages T₁ . . . T_n which point to it (i.e., are citations). The parameter d is a damping factor which can be set between 0 and 1. We usually set d to 0.85. There are more details about d in the next section. Also C(A) is defined as the number of links going out of page A. The PageRank of a page A is given as follows:

$$PR(A) = (1-d) + d (PR(T_1)/C(T_1) + \dots + PR(T_n)/C(T_n))$$

Note that the PageRanks form a probability distribution over web pages, so the sum of all web pages' PageRanks will be one. (p. 4)

There are some key observations from the previous mathematical expression that reflect what factors affect the PR of Page A. The PR of Page A is higher when:

1. There are many pages that link to it. The larger the set of pages T₁-T_n is, the larger the summation of their collective contributions.
2. Each of individual contributions of pages T₁-T_n to Page A's PR is higher when:
 - a. They have fewer outgoing links, represented by the variable C in the denominator of each term. This means the value of a link to Page A is lessened when the page that links to it has a large number of links to other pages.
 - b. They have high PR values themselves (a factor in the numerator of the each term in the summation).

However, the PR calculation also invites some questions. What is the *damping factor* shown by the variable *d*, and why is it usually set to 0.85? Even more puzzling, there is the apparent paradox of needing to know the PR of the pages that point to Page A before calculating its PR; how can those PRs be calculated if they too depend on calculating other PRs?

Of these questions, the damping factor issue is relatively straightforward. The variable *d* occurs in two places in this expression. It is in the first term of the formula where it is subtracted from 1; it occurs again when it is used in a product with the cumulative summation of all the PR terms. It is called the damping factor because when the summation of the terms is multiplied by *d*, it will reduce (or dampen) their total collective contribution. Its occurrence in the first term (1-*d*) has two effects: first, it means that even a page with no backlinks will have some PR and second, as noted in the quote from Brin and Page (1998), it creates a probability distribution, which means that the sum of all PRs is 1. The value 0.85 as the damping factor has been determined by the developers to optimize the PR process.

As for the problem of calculating a PR that depends on calculating other PR values first, Brin and Page (1998) explain it as follows: "PageRank or $PR(A)$ can be calculated using a simple iterative algorithm, and corresponds to the principal eigenvector of the normalized link matrix of the web" (p. 4). A discussion

of eigenvectors or Markov chains is not essential to a “big picture” view; the simplified answer is that the formula starts with an estimated initial PR value, and with not too many subsequent iterations of the calculation, the values trend to an acceptably useful PR value (Rogers, 2002).

PageRank has proved to be effective in relevance ranking of retrieval sets, but it is also susceptible to manipulation, as demonstrated by the creation of *link bombs*. A link bomb refers to a coordinated effort to mislead the PR algorithm by deliberately creating links from certain words to a targeted page of interest. This technique is attributed to Adam Mathes who created one in 2001 that placed a friend’s Web page as the top retrieval for the search “talentless hack” (Tatum, 2005). A well-known example of link manipulation resulted in a Google search using the words “miserable failure” placing President George W. Bush’s biography at the top of the retrieval set. Such manipulation does not take a large-scale effort; it is estimated that associating that result may have taken as few as 32 (BBC News, 2003), and certainly no more than several hundred, pages submitted to Google with links to the Bush page using anchor text with those terms (Sullivan, 2004). Google has since made changes to its algorithm to detect and prevent such manipulation of the PR rankings, but there is an ongoing cat and mouse game between Google and those who seek to manipulate it. To summarize, the main ideas in this discussion of link analysis are:

1. Citation analysis is useful in determining the importance and authority of sources and this principle can be adapted to the hypertext nature of the Web.
2. The hypertext structure of the Web can be exploited algorithmically to provide numeric rankings that factor into a determination of relevance and/or authority.
3. There are different models for link analysis, two of which are HITS and PR.
4. HITS is primarily about identifying authorities and the hubs that interconnect them.
5. The calculation of Google PR for pages in a retrieval set can be used in overall ranking. The PR of a given page is determined by both the number of pages that link to a page and the PR of those pages.

Google Revisions and SEO

As mentioned earlier, Google makes use of many signals in addition to Page Rank, and it makes hundreds of adjustments to its algorithms each year. Some of these changes are minor, but some are of more importance and are given names, such as Panda and more recently, Penguin (Google algorithm change history, 2013). Throughout its history, there has been a back and forth battle between Google and those who use an understanding of how it works to try to “game” the system—the link bombs described previously were an early example

of such abuse. Manipulating Google results is not just done as a practical joke. Google's AdSense business model means that attracting visitors and clicks is big business, and pages were often created to monetize keyword phrases rather than to provide useful content to the visitor, usually by building pages around high-value keywords that occurred numerous times throughout the page title, headers, or body. Google's algorithms can spot such unnatural word density and deems these pages "webspam." Google considers some optimization strategies acceptable and others as abusive (Webmaster guidelines, 2013). These techniques and Google's responses to them are relevant to website designers, searchers, and to the topic of semantic search that follows.

Panda was a major update to Google search that was designed to demote low-quality sites that do not provide useful original content and to promote the ranking of high-quality sites (Cutts, 2013). That refinement was in part aimed at reducing retrievals from *content farms*, which are sites with large amounts of textual content designed to be placed high in retrieval sets and gaining revenue-generating page views. With Panda, the main goal was to ensure that highly ranked sites represented high-quality content. Penguin, first released in 2012, is designed to identify sites that are spamming Google results but still ranking highly (Sullivan, 2013). With these updates, quality content from authoritative, trusted sites will be ranked higher. The 2013 Hummingbird algorithm represents a major change to Google's engine (Sullivan, 2013). This development is designed to enhance conversational search as well as to better manage complex queries and query relationships with Web documents (Gesenhues, 2013).

The notion of authority is of particular interest to this discussion—it was part of the HITS model discussed earlier in this chapter, and it is relevant to the discussion of semantic search that follows in the next section. The Penguin update refines the PageRank algorithm to emphasize the *quality* of the pages that link to yours; links from high-quality pages improve your rankings, while links from poor-quality sites are detrimental, even if the total number of back-links is high. Quality is not just measured by the presence of unique content, but also by semantic analysis that looks for the presence of niche vocabulary that would be employed by a content expert writing on a particular topic as well as considering authority conferred by reviews or other social signals.

SEMANTIC SEARCH

The idea of a semantic Web has been promoted by Web founder Tim Berners-Lee since 1999 when he suggested the next generation Web would be based on XML (Berners-Lee, 1999). Although that prediction has not come to fruition, more and more content of the Web is becoming part of a linked open data (LOD) cloud with associated semantic and descriptive metadata (Ruotsalo, 2012). The core idea of Web 1.0 is about document connections, Web 2.0 is about applications, and the semantic Web is about connecting data within these documents and applications at a more granular level. Semantic search is designed to take advantage of these data relationships. Fay and Sauers describe semantic search as "a new way of searching that takes advantage of connecting data" that relies on both metadata and linking data together (2012, p. 2).

Conceptually, semantic search depends on three interrelated components: a database of URIs, RDF, and an ontology library (Amerland, 2014). A URI, discussed in other parts of this text, might be the familiar URLs or URNs that identify a site or a document, but it might also be for a specific point of content within a document. RDF, an XML implementation described in Chapter 12, provides a set of rules that can map or translate data from one source to another without any loss of meaning. That translation is facilitated by an ontology library that collects and stores the various meanings of the data and concepts, their associations, and inference rules that can represent the knowledge of a domain. An example of an ontology used with RDF is the FOAF (Friend of a Friend) vocabulary used to describe people and their activities and their interactions (FOAF vocabulary specification, 2010).

A semantic Web and semantic search depend on metadata. Metadata can be added by the content creator, a third party, or can be algorithmically extracted or inferred from the raw content itself. RDF was the data model for metadata for the original conceptual view of the Semantic Web. Identifying entities (what something is about), attributes (their properties), and all the data interrelationships is key to a semantic Web—each entity, attribute, and relationship is potentially useful metadata, and the unique content identifier is a URI. RDF is a way to store connections among these nodes.

A number of technologies are available to facilitate the addition of metadata that in turn can support semantic search. As described in Chapter 8, HTML5 provides some standard semantic elements, and RDFa, discussed in Chapter 12, adds a set of markup attributes that can be embedded within HTML5, XHTML, and XML content markup to provide metadata (RDFa primer, 2013). The embedded data is in the form of attribute-value pairs that can be utilized by programs and search engines. RDFa attributes refine the meaning of the content or associate it with other nodes or vocabularies with a URI reference. For example, a simple H1 tag in HTML5 could be associated with the DC vocabulary title element by adding a property attribute as shown below:

<h1 property = “<http://purl.org/dc/terms/title>”>My title</h1>

This attribute addition does not alter the display but does provide machine-readable metadata about the nature of the header content, defining it not just as a generic header but also as a DC title. For content not associated with a specific HTML element, the generic SPAN tag can be used to add RDFa attribute metadata, allowing any content in a document to have machine-readable metadata associated with it.

Microdata and microformats each represent another way to further the use of embedded metadata. Microformats are built on existing standards such as XHTML (About microdata, 2013). Microdata is a similar, newer W3C format that is associated with HTML5 (Smith, 2010). To encourage its use and support their search engines, Google, Bing, and Yahoo launched Schema.org that provides various schemas for different types of Web content, such as creative works (books, music, recipes, etc.), events, persons, places, products, or reviews. The schemas identify “things” and hierarchies—for example, a place is a specific type of thing, and a business a specific type of place. Properties are defined for entities, and they can inherit properties from broader types. Tags or their attributes carry this

metadata that can then be extracted and used in semantic search. Examples of microdata are given in the discussion of Google Snippets below.

The implementations of semantic search by large-scale search engines also depends on “big data” and algorithms that can identify and store and translate relationship information, with the goal of understanding the “intent” of the searcher and the context of the terms in a query or question. Humans are very good at using contextual information to eliminate ambiguity and determine meaning. Consider the statement “the chickens are ready to eat.” If the context was a barbecue, it probably is to announce dinner is ready. If the context is a farmyard, it probably means it is time to scatter some grain to feed the chickens. Computers have long found this type of problem challenging, but large-scale search engines have huge amounts of data to analyze and use to identify or infer associations and relationships. Adding contextual information to data allows computers and software to do more for the searcher and yield better results. There has been an ongoing and growing interest in new applications of semantic search and the tools to support it such as location-based data and microformats/microdata to enrich content; examples include Twitter’s real-time search and Bing’s location-based map search (Fay & Sauers, 2012). An interesting example of a strictly semantic knowledge engine is WolframAlpha (<http://www.wolframalpha.com/>).

Another source of semantic data is the “social signal” of a user on the Web, which can in turn be used to assign a level of trust regarding their content. Your social signal is the collective effect of all your Web-related activities—commenting or responding to comments on websites or blogs, writing reviews, “liking” someone or something, or having your own blog or Twitter following (Amerland, 2014). Google uses this data to assign a “TrustRank” and uses it with other ranking signals in their algorithms to promote or demote sources to ensure the information provided is both relevant and accurate.

Google, Bing, and Yahoo are all using semantic search technologies. These search engines consider not just individual terms but the meaning and relationships of the search words together, treating the entirety of the question asked as well as semantic data from many sources to situate the context of the query. Google, Apple, and Microsoft Bing all also support voice search; Apple has Siri and Bing has released Natural Language Voice Search for its apps and latest Xbox. Semantic searching seeks to return results that predict the intention of the search. For example, a semantic search can expand the results for a city name search to include commonly needed information such as its location, climate, population, or other facts about it in addition to returning a page of links to items such as its official website (Oswald, 2012). Other sources of data such as your personal search history can also influence the query results.

A few major updates to Google’s algorithms that support this have been discussed so far in the context of SEO, but these refinements are not just to prevent bad actors from abusing the system—they also support improved searching through the development and use of semantically enhanced search. Google and its competitors Bing and Yahoo have sought to increase both the ease and the effectiveness of Internet search, and a few of these features are highlighted below.

Google's Knowledge Graph

Google's Knowledge Graph, released in 2012, seeks to make Google an answer engine rather than just a search engine; instead of returning links to sources that might answer a question, it tries to provide it directly in the response. After 15 years of storing and analyzing searches, Google has built a huge database of 570 million entities to support semantic search (Zaino, 2013). This data, combined with input from various social signals and fuzzy Boolean logic, allows Google to better anticipate the intent of a query. The process begins as soon as the first search term of a question is being entered—Google begins to make suggestions to complete the query with terms that have been previously associated with it. Those suggestions may differ over time or from user to user depending on if additional information from your personal search history, your Google account, or even your location is available to filter the pool of query suggestions or the search results presented. The Knowledge Graph is just that—a graph of interconnected entities and information about their attributes—it is all that cross-referenced information from across the Web that allows Google to generate lists of objects associated with a term (Amerland, 2014). Figure 15.3 is a screenshot of the Google “inside search” explanation of the Knowledge Graph showing the connections for a search for “da vinci.”



Figure 15.3 Screen shot from Google's Inside Search about Knowledge Graph.

An example of how Google has evolved from just providing links to providing direct answers to questions within the first response page is shown in Figure 15.4. The Knowledge Graph displays on the right side of the search results and following a link within it can bring up a “carousel” of related content displayed as a scrolling set of images at the top of the screen. The search was for musician Leo Kottke and the result page is not just a collection of links. It is divided vertically—the left side has expected links, but the right side has immediate information: a short biography, photo, a frame window with upcoming events, and a list of popular albums and songs. Clicking on the heading “albums” in the Knowledge Graph would bring up a carousel of album images at the top of the screen. Semantic search anticipates that I might be interested in a nearby concert and it uses my location to highlight one that is near my location.

As search engines seek to become “knowledge engines,” one caveat is that the answers presented may not be accurate; they are only as good as the sources they come from such as Wikipedia and Google Places. There are reports of significant errors, but Google allows users to correct these with the “Feedback/more info” link (Notess, 2013).

Google Snippets

One of the sources for some of the rich data Google search can utilize is the metadata being inserted to HTML with microformats or microdata. RDFa, microformats, and microdata are all ways that allow Web developers to “semanticize”

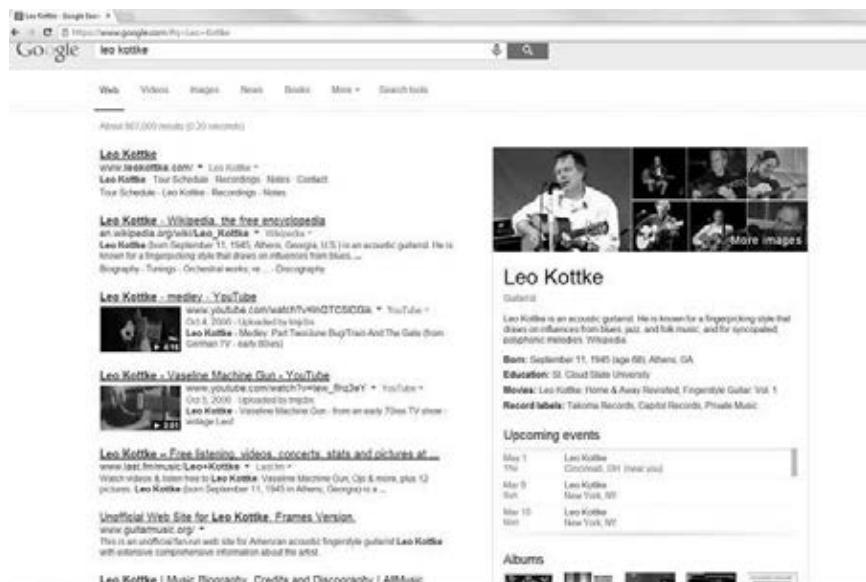


Figure 15.4 Screen shot of the Google Knowledge Graph for the search “Leo Kottke.”

their content to make it more findable and to enhance the display of information on the search results page. All are supported, but Google recommends the microdata approach in their documentation. Google Rich Snippets provides one way to enrich some common types of Web pages with metadata. Content such as pages related to events, directories, reviews, recipes, organization information, music, and videos are all excellent candidates for adding metadata in this way. Google's Webmaster Tools have many helpful sources on this topic such as a structured data testing tool and a data highlighter (About rich snippets, 2014).

The Schema.org site provides many schemas and examples for using microdata. For those who have mastered HTML, the markup needed for these enhancements is quite straightforward (the increasing use of these techniques is another good reason to learn HTML). Information about a DIV section can be provided with **itemscope** and **itemtype** attributes. Within that container, the itemprop attribute allows values such as names or titles. For example, a Web page about the movie *The Hobbit* could reference the movie schema and use these attributes as follows:

```
<div itemscope itemtype="http://schema.org/Movie">
<h1 itemprop="name">The Hobbit</h1>
<span itemprop="genre">Fantasy</span> . . .
```

Bing Smart Search

Bing is the current version of Microsoft's search engine that began as MSN Search. Bing also powers Yahoo search, although Yahoo retains its own interface. Bing's Smart Search is integrated into Windows 8.1. Bing's semantic search Snapshot display is similar to Google's Knowledge Graph; for example, the Leo Kottke search done on Bing has some different information but is very similar in appearance and data. These similarities in results led to a controversy in 2011 when a study reported Bing was copying Google's results, which Microsoft denied (Sullivan, 2011). That controversy aside, in the last few years Bing has become more competitive with Google in semantic search and has added similar features; for example, using location information and adding conversational voice search. Both search engines use a two-column result display and each has apps to support specialized functions and specialized search for shopping, maps, or recipes. Bing has modernized their start page with a full screen image and a scrollable image strip at the bottom of the screen that link to trending stories.

PEER-TO-PEER

P2P employs a distributed computing model for information seeking and represents an alternative to the large-scale, centralized indexes typical of most search engines. One implementation of peer-to-peer file sharing is the Gnutella software. Gnutella gained considerable notoriety with the Napster

music-sharing service, which after numerous legal challenges has become a fee-based service. Other similar services using this P2P model are Kazaa and LimeWire. Although these music distribution services have been one of the main uses of P2P, Gnutella P2P is an agnostic service, that is, the type of information requested or served does not matter (Ding, Nutanong, & Buyya, 2003). Therefore, P2P approaches are applicable to many types of file and information sharing, from simple text messaging and bookmark sharing to the shared computational activities of the “SETI at Home” project, which uses millions of personal computers to analyze radio telescope data for signs of extraterrestrial life.

In P2P, multiple computers running a common client can share tasks, messages, or search queries. A computer in the network sends a query for processing to the computers in its immediate “neighborhood” of participating systems, and each recipient performs a search of its local index. If the query is not answered, each computer in the initial neighborhood would pass it on to an ever-expanding network of participating systems. The number of hosts working on the search can become very large very quickly, and the larger and more diverse the network is, the higher the likelihood of finding an appropriate answer.

P2P information services are somewhat analogous to a party game where one person in a room can ask a question of the three people around them. If none of those three people know the answer, they can each pass the query to the three people around each of them. If none of those nine additional people knows the answer, each of the nine can then query three more people in an ever-widening pool of potential sources of an answer, until eventually someone who knows the answer sends it directly to the person who started the process. The success of the game depends on the nature of the query, the number of people in the room, and the knowledge of the individuals present. There are many distributed search engines based on P2P, and the O'Reilly Openp2p.com site lists more than two dozen of them, including the Clip2 Distributed Search Solutions for technical data and research aimed at Gnutella developers and users, and the Eikon prototype for distributed image search (O'Reilly Media, Inc., 2008).

CLUSTERING AND VISUALIZATION

The large retrieval sets returned for broad queries create the abundance problem alluded to earlier with the result that researchers are often drowning in information and starved for knowledge. One solution is the use of authority scores based on link analysis as previously described. However, even with good relevance ranking, retrieval sets are often large, and how the set is presented influences search success. A challenge for the user when a large set is retrieved is that the visual presentation of the set is often just a linear list of ranked results. Users often never see potentially excellent retrievals because they rarely scroll through more than the first few screens of results. Clustering technologies can provide visualization techniques that breakup a large set into a number of subsets representing useful categories. The clustering of results within visual displays can show the relationships among the categories. Sites of this type not only attempt to create a knowledge map of interrelated categories but

also can dynamically rearrange the display based on user input. Clustering is just one facet of the broader topic of data visualization, which seeks to present information clearly and efficiently in a graphical display. The power of such visual techniques is demonstrated by a simple pie chart used to explain budget data. Tools now exist to create visualizations for more complex data sets, including search results. An interesting example to explore is from the searchable collection of infographs and data visualizations found at the [visual.ly](#) website.

A number of search engines have experimented with ways to cluster results visually or organize them into named category folders of similar retrievals. Some of the early clustering sites such as Northern Light, Vivisimo, WebBrain, and KartOO have gone private, disappeared, or discontinued their result set clustering options. One example still available is the Quintura search engine for kids (<http://quinturakids.com/>), which combines the functions of search and a directory within a graphical “cloud” of terms to assist in organizing and refining a search. A related topic is the use of iconography to visually present search results. There are a number of search engines using this approach. Redz.com is a general search engine that allows separate searching of the Web, video, and images. The more site-specific oSkope.com visual search engine can only be used to search within Amazon, eBay, YouTube, Flickr, and fotoilia. Both services present search results as a set of thumbnail-type graphics. Visualization tools are also available to analyze social media networks or the link structure of pages on the Web. NodeXL (<http://nodexl.codeplex.com/>) is an open-source plugin for Excel to create network graphs. Library catalogs also have experimented with concept mapping and visualization features to assist with subject exploration. These displays can present search results using images of book covers and show item details in a popup window. Visualization and topic mapping represent a promising approach in the battle against information overload.

SMART AGENTS, PERSONALIZATION, AND PRIVACY

The idea of “smart agent” software has been around for some time; generally, this refers to software that learns about your interests and can automatically do your bidding by running in the background, searching for new information of interest, and pushing it to you. Ideally, such a program could learn not just from you and your actions but also from other agent programs as well. Semantic search combined with personalized search and geolocation data have resulted in apps that are beginning to approximate this kind of machine learning. Search engines and other Web technologies such as RSS also allow much more sophisticated ways to anticipate client interests and “push” the information to them on a regular basis.

Personalization is also possible with client-server cookie exchanges that serve as a simple form automated information update. As described in Chapter 5, cookies serve as an HTTP “state object” used to maintain information about a transaction that is retrievable by the server that placed them. When you log into your account in Amazon, the cookie exchange identifies you to the server, and you are welcomed with a list of suggestions of items thought to be of interest to you. These suggestions come from transactional information;

consequently, it is not always very “smart” in its recommendations. For instance, if you recently bought gift books for others, the system assumes they reflect your personal interest areas for future shopping. Mobile apps do not use desktop cookies but instead use a similar strategy involving a token set by the app that can be retrieved later; because of their similar function, these tokens are often also called a cookie (Perez, 2013). Google stores personal preferences in a PREF cookie. The use of these cookies continues to raise privacy concerns, especially in light of the 2013 Snowden leaks that revealed the NSA’s secret use of PREF cookies and tracking data (Soltani et al., 2013).

Personalization techniques have implications for Web search, and many services allow customization of search engine portals with a user profile such as the My Yahoo portal (<http://my.yahoo.com/>). However, the prevalence of mobile devices and apps options for personalizing the Web experience has led to the demise of some of these personalized sites; for example, the personally configured Web page option from Google (iGoogle) for the desktop is no longer supported. Google’s new personalized option is Google Now, an intelligent personal assistant mobile app that is sometimes referred to as a “psychic app” because it is designed to use all they know about you and your location to anticipate your information needs and offer unsolicited suggestions based on your activities (Reed, 2012).

Search engines can push information to a user based on a profile or automatically filter future search results based on known user interests, search history, and locations. Early on, many users were concerned with Google’s retention of persistent identifiers, such as IP addresses and User IDs (EPIC, 2008). Google has confirmed that it collects information from users across all its various services (Kang, 2012). This information from activities tied to your account and your personal search history can be used to enhance and personalize search.

In addition to collecting these data to yield personalized search results, search engines could also view user profile information as a potential income stream, so there must be clear policies in place that control the use of such information. Information about user searches and its potential misuse has become a controversial political issue; many governments, including those of the United States and China, have sought information about searches done by users of these services (Associated Press, 2006; Klein, 2008). The revelations of extensive data collection within the United States and abroad by the NSA have raised many concerns among privacy advocates. There are search engines specifically designed to maximize user privacy such as DuckDuckGo.com and ixquick.com. Another search engine with strong privacy policies and a commitment to keeping spam out of results is Blekko.com.

SPECIALIZED SEARCH ENGINES

While the general purpose search engine is often the starting point for an exploratory search, a specialized search engine may offer a more efficient strategy for some information needs. Scheeren (2011) lists many search engines specifically designed for invisible Web content. In addition to the invisible Web, there are many niche services, a few of which are mentioned below.

- YouTube (youtube.com) is an enormously popular site that is also a good example of a highly successful vertical search engine for video content; it is the second largest search engine after its Google parent (Amerland, 2013).
- Facebook Graph Search: Facebook is one of the largest social networking sites and its search engine allows users to search for people, places, posts, status updates, and comments. Searches can be refined by time frame or location.
- TinEye (tineye.com) is an interesting reverse image search engine that allows images to be uploaded or pointed to and then tries to find the page associated with it. However, it cannot analyze the content of the image itself.
- Google image search is an example of a CBIR engine that uses an uploaded image or an image URL as the query. The search results include links related to the content of the image as well as to pages that include matching images.
- Midomi (<http://www.midomi.com/>) and Shazam were mentioned in Chapter 14 as examples of content-based media retrieval. Midomi is a search engine that can identify music by singing or humming a tune and the Shazam music identifier app can tag a song being played on your device by creating a “fingerprint” spectrogram and then comparing it to a database catalog of such fingerprints (Jacobs, 2010; Wang, n.d.).

COMPARISON WITH TRADITIONAL ONLINE SERVICES

It is informative to conclude this discussion of Internet search with a comparison of free Web services with the traditional and highly structured online information systems used in libraries, such as Dialog and Lexis-Nexis. Dialog, now combined with DATASTAR and owned by ProQuest, started in the 1970s and was one of the first online database services, providing access to more than 600 separate databases (Dialog Database Selection Guide, 2007). Prior to the Internet, part of the cost was also for the dial-up telecommunication fee to make the connection but Internet telnet and HTTP access eliminated those separate connection fees. Dialog ensures their databases have extensive quality control and they provide database information sheets (referred to as “Bluesheets” because they were originally printed on blue paper) and a consistent search environment across all databases using either a single command-based or a graphical Web interface. Both interfaces allow for powerful, controlled searching of multiple indexes along with advanced Boolean set manipulation. In addition to natural language keyword searching with KWIC (keyword in context) display, many of these database indexes utilize a controlled vocabulary with an online thesaurus that facilitates term selection. This database service provides access to content through an interface that can offer many advanced search options that are typically not available with general search engines and are designed for professional searching. Table 15.1 compares some features of these premium services with general search engines.

TABLE 15.1 Comparison of Internet Search with Dialog

Feature	Internet Search Engine	Dialog
Query Terms	Typically natural language keywords	Both keyword and controlled vocabularies
Indexing	Typically a single index	Multiple indexes allow for more search options (author, personal name, descriptor, etc.)
Interface	Multiple approaches; dependent on the search engine	Single command or Web-based interface; common syntax across all databases
Advanced Search	Limited advanced search options that differ across many search engines	Many advanced search options (proximity searching, set manipulation, field restrictors, etc.)
Database	Little to none	Rigorous
Quality Control		
Database Organization	Little to none	Catalogued and described
Database Selection	Few options, perhaps restriction by domain or resource type	Hundreds of specialized databases with documentation and selection guidance; search single, multiple, or all databases.
Documentation	Little, often difficult to find	Complete; printed and Web based

Although this service is aimed solely at information professionals with highly specific needs, there are also hundreds of similar high-quality subscription databases available to all library users. For example, EBSCOhost is a commonly used database aggregator aimed at the end user seeking access to full-text journals, newspapers, and magazines, eBooks, and other proprietary citation databases. EBSCOhost revised its interface to produce EBSCO host 2.0 to incorporate features that users of Web search expect, such as a clean interface, a simple basic search option, and advanced search options when needed.

A personalization feature called MyEBSCOhost allows users to save searches and setup automatic alerts for searches or journals.

Free Web-based searching has changed how people look for information and has given them the impression that every kind of information they seek is easily found with a Web search engine alone. Web search is extremely efficient for looking up information, but it often cannot support deep academic research; even though specialized search engines such as Google Scholar allow searching of such content, the full-text is often not free. Libraries do offer their users free access to many of these sources. The challenges are to raise awareness of what they offer and to make them competitive with search engines in terms of ease of use and utility. There are key differences between search engines and these databases in what they offer, their default search assumptions, and the search features they have. Databases do offer field specific and filtering options. However, users of search engines have become accustomed to automatic features such as implied AND among terms, fuzzy Boolean that does not require exact matches, term stemming, and relevance ranking of results. Many of these features are not offered by traditional catalogs and databases but are being added to many next generation discovery tools.

The issues explored in this chapter as well as the nature of the information gaps on the Web make it clear that the fee-based online databases or subscription services found in libraries still have an important place for many types of information seeking. In addition to full text of subscription sources, they provide access to specialized information not otherwise available, such as patent and trademark databases or legal information. The databases, rich feature set, and unified search interface available from services such as EBSCOhost, Pro-Quest, Dialog, Lexis-Nexis, or Westlaw give both the expert searcher and the end user access to both content and search options that are not possible with a general Web search.

SUMMARY

The ease and power of free Internet search have transformed both the Internet itself and how it is used for information seeking. During its relatively short history, Internet searching has gone from a few searchers using the simple VERONICA index of Gopher servers to the billions of daily queries accessing the enormous Google index. The different types of queries people pose result in the dual problems of scarcity and abundance, and much of Google's initial success was due to the application of link analysis techniques to these problems. Large retrieval sets make finding the best items within the returned set an added challenge. The early promises of the Semantic Web are beginning to come to fruition with the increased application and use of metadata as well as algorithms to extract relationships and meaning from the "big data" held by search engines. Semantic search has emerged as a way to overcome many of the early limitations of Internet search both in how queries are interpreted and in finding meaningful results, often in the form of a direct answer instead of a page of links to sources that may have an answer. Categorization and/or visualization techniques that organize the items into meaningful subsets can enhance their usability. Personalization of search and the vast amount of data

from social media and Web 2.0 applications that comprise one's "social signal" can improve search effectiveness but also pose potential privacy issues. Natural language queries and voice search systems such as Siri demonstrate how semantic search is beginning to emulate more intelligent human-computer conversations. A comparison of how the open Web search stacks up against searching more structured information systems such as Dialog, EBSCOhost, and the many subscription based library databases leads to the conclusion that although Web search continues to bring more free information to the public, there are still some information needs that are better satisfied with these fee-based or licensed systems.

REFERENCES

- About microformats. (n.d.). Retrieved January 27, 2014, from <http://microformats.org/about>.
- About rich snippets and structured data. (2014). Retrieved January 5, 2014, from <https://support.google.com/webmasters/answer/99170?hl=en>.
- Adamo, Stephanie. (2013, January 11). comScore releases December 2012 U.S. search engine rankings. Retrieved July 22, 2013, from http://www.comscore.com/Insights/Press_Releases/2013/1/comScore_Releases_December_2012_U.S._Search_Engine_Rankings.
- Amerland, D. (2014). *Google semantic search: Search engine optimization (SEO) techniques that gets your company more traffic, increases brand impact and amplifies your online presence* (1st ed.). Indianapolis, IN: Que.
- Associated Press. (2006, January 20). Google won't hand over files. Retrieved October 1, 2007, from <http://www.wired.com/politics/law/news/2006/01/70055>.
- Barroso, L.A., Dean, J., & Hölzle, U. (2003). Web search for a planet: The Google cluster architecture. *Institute of Electrical and Electronics Engineers (IEEE) Micro*, 23(2), 22–28.
- Battelle, J. (2005). *The search*. New York: Penguin Group.
- BBC News. (2003, December 7). "Miserable failure" links to Bush. Retrieved May 10, 2008, from <http://news.bbc.co.uk/2/hi/americas/3298443.stm>.
- Bergman, M. K. (2001, September 24). The "Deep" Web: Surfacing hidden value. Retrieved October 1, 2007, from <http://www.brightplanet.com/images/stories/pdf/deepwebwhitepaper.pdf>.
- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.
- Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic Web. *Scientific American*, 284(5), 10.
- Bort, Julie. (2013, May 13). Nearly 500 million searches a day are for things Google has never heard of. Retrieved July 19, 2013, from <http://www.businessinsider.com/500m-things-google-has-never-heard-of-2013-5>.
- Boutell.com. (2007, February 15). How many websites are there? Retrieved October 1, 2007, from <http://www.boutell.com/newfaq/misc/sizeofweb.html>.
- Brin, S., & Page, L. (1998). *The anatomy of a large-scale search hypertextual Web search engine*. Paper presented at the Proceedings of Seventh World Wide Web Conference, Brisbane, Australia.
- Broder, A., Fontoura, M., Josifovski, V., & Riedel, L. (2007). A semantic approach to contextual advertising. *ACM SIGIR International Conference on Research and Development in Information Retrieval*, 559–566. New York, NY: ACM Press.

- Chakrabarti, S., Dom, B., Kumar, S. R., Raghavan, P., Rajagopalan, S., Tomkins, A., et al. (1999). Hypersearching the Web. *Scientific American*, 280(6), 54–61.
- Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Gruher, R. E. (2006). Bigtable: A distributed storage system for structured data. Retrieved December 14, 2013, from <http://static.googleusercontent.com/media/research.google.com/en/us/archive/bigtable-osdi06.pdf>.
- Cheong, F. C. (1996). *Internet agents: Spiders, wanderers, brokers, and bots*. Indianapolis, IN: New Riders Publishing.
- Croft, W. Bruce, Metzler, Donald, & Strohman, Trevor. (2010). *Search engines: Information retrieval in practice*. Boston, MA: Addison-Wesley.
- Cutts, Matt. (2013). About search. Retrieved July 22, 2013, from <http://www.google.com/competition/howgooglesearchworks.html>.
- de Kunder, M. (September 2008). The size of the World Wide Web. Retrieved September 21, 2008, from <http://www.worldwidewebsize.com>.
- Devine, Jane, & Egger-Sider, Francine. (2009). *Going beyond Google: The invisible Web in learning and teaching*. New York: Neal-Schuman Publishers, Inc.
- Dialog® Database Selection Guide. (2007). Retrieved September 18, 2013, from http://support.dialog.com/techdocs/co018002mi_dlg_dbselguide.pdf.
- Ding, C. H., Nutanong, S., & Buyya, R. (2003). *Peer-to-peer networks for content sharing*. Melbourne, Australia: University of Melbourne.
- Dogpile. (April 2007). Different engines, different results. Retrieved June 1, 2008, from <http://www.infospaceinc.com/onlineprod/Overlap-DifferentEnginesDifferentResults.pdf>.
- Dragland, Åse. (2013). Big data—for better or worse. Retrieved September 29, 2013 from <http://www.sintef.no/home/Press-Room/Research-News/Big-Data—for-better-or-worse/>
- Efros, A. (2003). Data-driven texture and motion. In *University of Washington Computer Science and Engineering colloquia*. [Video file]. Retrieved September 28, 2008, from <http://www.researchchannel.org/prog/displayevent.aspx?rID=3244&fID=345>.
- Electronic Privacy Information Center (EPIC). (2008, May 21). Congressman Barton urges scrutiny of Google's privacy practices. Retrieved June 1, 2008, from <http://epic.org>.
- Fay, R. M., & Sauers, M. (2012). *Semantic Web technologies and social searching for librarians*. Chicago, IL: ALA TechSource.
- FOAF vocabulary specification 0.98. (2010, August 9). Retrieved January 4, 2014, from <http://xmlns.com/foaf/spec/>
- Garfield, E. (1972). Citation analysis as a tool in journal evaluation. *Science*, 178(4060), 471–479.
- Gesenhues, A. (2013, September 30). Google's hummingbird takes flight: SEOs give insight on Google's new algorithm. Retrieved December 30, 2013, from <http://searchengineland.com/hummingbird-has-the-industry-flapping-its-wings-in-excitement-reactions-from-seo-experts-on-googles-new-algorithm-173030>.
- Ghemawat, S., Gobioff, H., & Leung, S.-T. (2003, December). *The Google file system*. Paper presented at the Proceedings of the 19th ACM Symposium on Operating Systems Principles, Bolton Landing, New York.
- Gil, P. (August 2013). What is “the invisible Web”? Retrieved September 29, 2013, from <http://netforbeginners.about.com/cs/secondaryweb1/a/secondaryweb.htm>.
- Goodwin, Danny. (2013, January 15). Google Search market share slips as Bing & Yahoo gain. Retrieved July 22, 2013, from <http://searchenginewatch.com/article/2236637/Google-Search-Market-Share-Slips-as-Bing-Yahoo-Gain>.

- Google Algorithm change history. (2013). Retrieved December 12, 2013, from <http://moz.com/google-algorithm-change>.
- Gulli, A., & Signorini, A. (2005). *The indexable Web is more than 11.5 billion pages*. Paper presented at the WWW 2005, Chiba, Japan.
- Hawking, D., & Zobel, J. (2007). Does topic metadata help with Web search? *JASIST*, 58(5), 613–626.
- Hock, Randolph. (2013). *The extreme searcher's Internet handbook: A guide for the serious searcher* (4th ed.). Medford, NJ: CyberAge Books.
- Internet world users by language: Top 10 languages. (2011, May 31). Retrieved July 10, 2013, from <http://www.internetworldstats.com/stats7.htm>.
- Jacobs, Bryan. (2010, September 24). How Shazam works to identify (nearly) every song you throw at it. Retrieved July 19, 2013, from <http://gizmodo.com/5647458/how-shazam-works-to-identify-nearly-every-song-you-throw-at-it>.
- Jeanneney, J.-N. (2006). *Google and the myth of universal knowledge*. Trans. T. L. Fagan. Chicago, IL: University of Chicago Press.
- Kahle, B. (1997). Preserving the Internet. *Scientific American*, 276(3), 82–84.
- Kang, Cecilia. (2012, January 24). Google announces privacy changes across products; users can't opt out. Retrieved July 19, 2013, from http://www.washingtonpost.com/business/economy/google-tracks-consumers-across-products-users-cant-opt-out/2012/01/24/gIQArgJHOQ_story.html.
- Klein, N. (2008, May 29). China's all-seeing eye: With the help of U.S. defense contractors, China is building the prototype for a high-tech police state. Retrieved May 30, 2008, from http://www.rollingstone.com/politics/story/20797485/chinas_allseeing_eye.
- Kleinberg, J. M. (1999). Authoritative sources in a hypertext environment. *Journal of the ACM*, 46(5), 604–632.
- Langville, A. N. (2005). The linear algebra behind search engines. Retrieved October 15, 2007, from <http://mathdl.maa.org/mathDL/4/?pa=content&sa=viewDocument&nodeId=636>.
- Langville, A. N., & Meyer, C. D. (2006). *Google's PageRank and beyond: The science of search engine rankings*. Princeton, NJ: Princeton University Press.
- Lawrence, S., & Giles, C. L. (1998). Searching the World Wide Web. *Science*, 280(5360), 98–100.
- Lawrence, S., & Giles, C. L. (1999). Accessibility of information on the Web. *Nature*, 400(6740), 107–109.
- Lawrence, S., Coetzee, F., Glover, E., Flake, G., Pennock, D., Krovetz, B., et al. (2000). *Persistence of information on the web: Analyzing citations contained in research articles*. Paper presented at the Conference on Information and Knowledge Management, McLean, Virginia.
- McGee, Matt (2013, May 15). Bing rises above 17% search market share as Google slips [comScore]. Retrieved July 19, 2013, from <http://searchengineland.com/bing-rises-above-17-search-market-share-as-google-slips-comscore-159746>.
- Miller, Rich. (2011, August 1). Google uses about 900,000 servers. Retrieved September 22, 2013, from <http://www.datacenterknowledge.com/archives/2011/08/01/report-google-uses-about-900000-servers/>
- Mills, E. (2005, October 21). Google shares soar on hearty revenue report. Retrieved October 1, 2007, from http://news.com.com/Google+revenue+nearly+doubles/2100-1030_3-5905127.html.
- More than one third of web pages are pornographic. (2010, June 16). Press release. Retrieved July 18, 2013, from <http://www.optenet.com/en-us/new.asp?id=270>.

- Mowshowitz, A., & Kawaguchi, A. (2002). Bias on the Web. *Communications of the ACM*, 45(9), 56–60.
- Notess, G. R. (2002, March 6). Little overlap despite growth! Retrieved October 1, 2007, from <http://www.searchengineshowdown.com/statistics/overlap.shtml>.
- Notess, G. (2013). Search engine to knowledge engine? *Online Searcher*, 37(4), 3.
- Online Computer Library Center (OCLC). (2002). Web characterization. Retrieved October 1, 2007, from <http://www.oclc.org/research/projects/archive/wcp>.
- O'Reilly Media, Inc. (2008). Distributed search engines. Retrieved October 1, 2007, from <http://www.openp2p.com/pub/t/74>.
- Oswald, Ed. (2012, May 9). Demystifying semantic search. Retrieved July 19, 2013, from <http://www.extremetech.com/computing/123599-demystifying-semantic-search>.
- Perez, Sarah. (2013, February 25). Apple rejecting apps using cookie-tracking methods, signaling push to its own ad identifier technology is now underway. Retrieved September 26, 2013, from <http://techcrunch.com/2013/02/25/apple-rejecting-apps-using-cookie-tracking-methods-signaling-push-to-its-own-ad-identifier-technology-is-now-underway/>
- Pfeifer, R., & Bongard, J.C. (2007). *How the body shapes the way we think*. Cambridge: MIT Press.
- Purcell, Kristen, Brenner, Joanna, & Rainie, Lee (2012, March 9). Search engine use 2012. Retrieved September 11, 2013, from <http://www.pewinternet.org/Reports/2012/Search-Engine-Use-2012.aspx>.
- RDFa 1.1 primer—Second edition: Rich structured data markup for Web documents. (2013, August 22). Retrieved January 4, 2014, from <http://www.w3.org/TR/xhtml-rdfa-primer>.
- Reed, B. (2012, September 28). How Google wants to make Google Now a psychic stalker version of Siri. Retrieved December 26, 2013, from <http://bgr.com/2012/09/28/google-now-features-preemptive-answers/>
- Rogers, I. (2002). The Google PageRank algorithm and how it works. *Google PageRank whitepaper*. Retrieved October 10, 2007, from <http://www.ianrogers.net/google-page-rank>.
- Ruotsalo, T. (2012). Domain specific data retrieval on the semantic Web. In E. Simperl, P. Cimiano, A. Polleres, O. Corcho, & V. Presutti (Eds.), *The semantic Web: Research and applications* (vol. 7295, pp. 422–436). Berlin, Heidelberg: Springer.
- Ruvolo, Julie. (2011, September 7). How much of the Internet is actually for porn. Retrieved July 18, 2013, from <http://www.forbes.com/sites/julieruvolo/2011/09/07/how-much-of-the-internet-is-actually-for-porn/>
- Scheeren, William O. (2012). *The hidden Web: A sourcebook*. Santa Barbara, CA: Libraries Unlimited, an imprint of ABC-CLIO, LLC.
- Singhal, Amit, & Cutts, Matt. (2011, February 24). Finding more high-quality sites in search. Retrieved July 22, 2013, from <http://googleblog.blogspot.com/2011/02/finding-more-high-quality-sites-in.html>.
- Smalera, P. (2007, September). Google's secret formula. *Conde Nast Portfolio*, 136–137.
- Smith, C. (2010, October 12). What's best: Microformats, RDFa, or micro data? Retrieved January 27, 2014, from <http://www.semclubhouse.com/micro-formats-rdfa-or-micro-data/>
- Soltani, Ashkan, Peterson, Andrea, & Gellman, Barton. (2013, December 10). NSA uses Google cookies to pinpoint targets for hacking. Retrieved January 5, 2014, from <http://www.washingtonpost.com/blogs/the-switch/wp/2013/12/10/nsa-uses-google-cookies-to-pinpoint-targets-for-hacking/>

- Sullivan, D. (2004, January 6). Google's (and Inktomi's) miserable failure. Retrieved May 1, 2008, from <http://searchenginewatch.com/showPage.html?page=3296101>.
- Sullivan, D. (2005, January 28). Search engine sizes. Retrieved October 1, 2007, from <http://searchenginewatch.com/showPage.html?page=2156481#trend>.
- Sullivan, D. (2011, February 1). Google: Bing is cheating, copying our search results. Retrieved December 14, 2013, from <http://searchengineland.com/google-bing-is-cheating-copying-our-search-results-62914>.
- Sullivan, D. (2013, October 4). Penguin 5, with the Penguin 2.1 spam-filtering algorithm, is now live. Retrieved December 12, 2013, from <http://searchengineland.com/penguin-2-1-and-5-live-173632>.
- Tatum, C. (2005). Deconstructing Google bombs: A breach of symbolic power or just a goofy prank? *First Monday*, 10(10).
- Valenza, Joyce. (2013, July 15). Qwant a one-page, multilingual search aggregator? Retrieved July 19, 2013, from <http://blogs.slj.com/neverending/search/2013/07/15/qwant-a-one-page-multilingual-search-aggregator/>
- Vidmar, D., & Anderson, C. (2002). History of Internet search tools. In A. Kent & C. Hall (Eds.), *Encyclopedia of library and information science* (vol. 71, pp. 146–162). New York: Marcel Dekker, Inc.
- W3Techs. (2013, July 18). Usage of content languages for websites. Retrieved July 18, 2013, from http://w3techs.com/technologies/overview/content_language/all.
- Wang, Avery Li-Chun. (n.d.). An industrial-strength audio search algorithm. Retrieved July 19, 2013 from <http://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf>.
- Ward, Mark. (2013, June 30). Web porn: Just how much is there? Retrieved July 18, 2013, from <http://www.bbc.co.uk/news/technology-23030090>.
- Webmaster guidelines. (2013). Retrieved December 30, 2013, from <https://support.google.com/webmasters/answer/35769?hl=en>.
- Zaino, J. (2013, May 15). Talking the talk—And walking the walk—About the beauty of search at Google I/O. Retrieved January 4, 2014, from <https://semanticweb.com/tag/amit-singhal>.
- Zillman, Marcus P. (2006, June 10). Features—Deep Web research 2006. Retrieved July 15, 2013, from <http://www.llrx.com/features/deepweb2006.htm>

Libraries and the Internet: Learning from the Past, Exploring the Future

The introduction to this text asserted that in many ways, the Internet has changed everything. While acknowledging this is perhaps an overstatement in the broader scheme of human history, it does seem reasonable to make this assertion in the context of libraries and their services. However, the predicted transformative power of the Internet did not begin to be realized until the emergence of the Web in the 1990s. By the end of that decade, the information super highway metaphor of the pre-Web Internet and its focus on “getting connected” had given way to the power of hypertext and the search capabilities of the Web. In hindsight, it seems it was the Web, and not just the Internet, that changed the world. In the decade following its inception, the notion of Web 2.0 emerged, and almost as soon as it was announced, predictions of the inevitable evolution to Web 3.0 and beyond were made. Now more than a decade old, the term “Web 2.0” has become somewhat passé, but the participatory, collaborative communities it enabled continue to generate both hyperbole and genuine excitement. Social media and the mobile Web open new opportunities for the Web, regardless of the name or number given to it. When did Web 1.0 become Web 2.0? Obviously, this was not a discrete-time event; there was no announcement that Web 1.9 would rollover to Web 2.0 at midnight one day. In that sense, Web 1.0 coexists with Web 2.0, Web 3.0, and, very likely, Web 4.0, however that will be defined, if we have not become too tired of using such designations by then. The challenge that still confronts us is to identify and distinguish the trends and technologies that will form the next framework for library services from those technologies that are merely passing fads. Each phase of technological advancement is often announced as a revolution. Some will be genuine game changers, some will not last, and others more accurately represent an ongoing technological evolution as opposed to a revolutionary break with the past. Even though this nomenclature has lost some of its relevance,

these distinctions are nonetheless familiar to everyone and they will be used as a convenient way to organize parts of this discussion. The idea of 2.0 does have a powerful brand identity, a meme generically representing something modern, new, and improved. Web 2.0 supports or leads to business 2.0 or library 2.0, albeit with the same limitations associated with such a naming convention. Any name applied to a broad and complex mix of technologies and people would have limitations. Nevertheless, whether or not the 2.0 label is appropriate for the Web or for libraries, the participation element associated with it is central to its success and its future iterations regardless of how they will be described.

The transformative energy of Web 2.0 is not derived solely from the collaborative human activities it enables, but also from the morphing of the World Wide Web into the “World Wide Computer.” Christensen (2003) describes how innovations can become “disruptive technologies,” unleashing forces and opportunities that radically reshape the environment. He describes how entire industries have lost their market leadership by failing to recognize “the next great wave.” The cumulative changes of Web 1.0 combined with Web 2.0 and mobile applications represent such a disruptive technology, and libraries have not been immune to the forces Christensen describes. This adds some urgency to the dilemma of sorting out important trends from fads and presents the challenge of adapting to change while remaining true to the long-standing core values of librarianship.

Much of the perspective of this text is grounded in general Internet and Web 1.0 technologies, which remain relevant to the repositioning of the Web. While Web 2.0 allows much greater participation by those lacking the requisite technology expertise of earlier Internet activities, there is still a need for this expertise in the information professions. The backend technologies, databases, scripts, widgets, metadata, mashups, apps, and CMSs used in Web 2.0 still require substantive technology knowledge and skills by those seeking to implement and advance them. Those with a deeper understanding of how these technologies work will continue to advance the usability framework for others. One of the powerful outcomes of the developing Web 2.0 technologies is that by removing technology barriers it allows for expanded participation by those previously kept away by a lack of technical or design knowledge.

This concluding chapter is an opportunity to consider how the Internet, Web, Web 2.0, social media, and mobile technologies have affected the roles of librarians and libraries. The new roles and cultural shifts wrought by Web 2.0 gave rise to the Library 2.0 model, which continues to evolve. These shifts are both exciting and disruptive as information professionals must reevaluate, and possibly relinquish their traditional gatekeeper role and embrace more participatory models. The technologies that enable Web 2.0 have been addressed in other parts of this text; RSS, blogs, podcasts, wikis, apps, and social media tools are discussed in the chapters on TCP/IP, Internet protocols, Web markup languages and programming, XML, Internet content, and Internet IR.

In addition to the technologies traditionally associated with Web 2.0, other important developments that have been included in this second edition offer new opportunities and challenges to libraries. Open-source software options, cloud and grid computing, the social media juggernaut, and the emerging dominance of mobile devices as the main tool for communication and Internet access are all affecting society, libraries, and the information professions. These

TABLE 16.1 A Summary of Internet Periods and Their Descriptions, Protocols, and Common Applications in Libraries

Internet Period	Descriptors	Technologies	Applications
Pre-Web (1969–1992)	<ul style="list-style-type: none"> • Command based • Textual • Store and forward messaging 	<ul style="list-style-type: none"> • Telnet • Gopher • Email • SLIP/PPP 	<ul style="list-style-type: none"> • Connect to remote catalogs and databases • Email reference • Discussion lists
Web 1.0 (1992–present)	<ul style="list-style-type: none"> • GUI • Multimedia • Hypertext • Search/Pull/Static • Information silos • IM 	<ul style="list-style-type: none"> • Browsers • Audio streaming • User-centered links • Google • HTTP file delivery 	<ul style="list-style-type: none"> • Information Web pages • Web delivery of databases/OPAC but no integrated one-search
Web 2.0 (~2004–present)	<ul style="list-style-type: none"> • Collaborative • Social • Interactive • Functional hybridization (Mashups) • Shared tagging 	<ul style="list-style-type: none"> • Blogs • Wikis • Podcasts • Del.icio.us • Flickr • YouTube • Second Life 	<ul style="list-style-type: none"> • Blogs for internal/external audience • Collaborative knowledge sharing • Virtual library
Web 3.0 (Now–future)	<ul style="list-style-type: none"> • Semantic • “Intelligent” Web • Metadata • XML based 	<ul style="list-style-type: none"> • XML and JSON • EAD • RDF, RDFa, Microformats • MODS/METS 	<ul style="list-style-type: none"> • Data interchange • Archival finding aids • Metadata interoperability and embedded metadata

technologies are considered holistically in this chapter through the Web 2.0/Library 2.0 lens. Early on in this text, the mnemonic of the “three Cs,” for *computers, connections, and a common language*, was used to define the Internet. A similar “Rule of Cs” emerged to describe the Web/Library 2.0 shift; Jenny Levine describes the “4 Cs” of Web 2.0 as “*collaboration, community, commons, and conversation*” (Stephens, 2007, p. 7). Now some invoke another new “Rule of Cs” for social media, describing it as being about *connections, communities, content, consumer-centric storytelling, and conversations* (Turnau, 2012).

Web 2.0 continues to command a tremendous amount of attention and because of its importance to libraries and their services, much of this chapter is devoted to it. This discussion begins with a brief look at the history of the Internet in libraries as a prelude to discussing current trends and applications. Table 16.1 summarizes the defining properties of the Internet over time.

THE INTERNET: PRE-WEB

To the Millennials who have grown up with the Web, the early uses of the Internet must seem quite primitive and unexciting. The role of “Internet cheerleader” was a tough sell in the pre-Web environment; the command-based view of the original Internet protocols was neither user friendly nor intuitive, and there was a steep learning curve even for the tech-savvy user. Telnet connections to catalogs, OCLC, and database services such as Dialog were useful to librarians and even some patrons, but they did not engage a mass audience. Connectivity was often problematic as there were few Internet service providers, and broadband was only available in the form of T1 or fractional T1 lines, which were often prohibitively expensive. Libraries debated the necessity and desirability of providing general Internet access to the public, as well as the many policy and use issues it introduced. Even those who favored providing Internet access as a key service often found they did not have the computers, infrastructure, or in-house expertise to deliver it; consequently, Internet access for the public was not common in libraries during this period.

WEB 1.0

As described in other parts of this text, the emergence of the Web changed the Internet environment in significant ways and drove a reinforcing cycle of increased access and use. The core idea of Web 1.0 is hypertext linking—links to other resources embedded within the referring source make it unnecessary for the user to know anything about the remote server or need a login to access that linked resource. Viewed with a graphical browser, the Web was an immediate success, and traffic grew quickly; PC sales increased as did demand for bandwidth as the Web captured the attention of businesses and the public. The resulting dot com boom of the late 1990s drove huge investments in infrastructure that have made Internet access and increased bandwidth available throughout much of the United States. By the end of that decade, Internet access was a well-established service in libraries, schools, and most homes. U.S. Census data show the number of homes with Internet access in the United

States went from 18 percent in 1997 to 54.3 percent in 2003 (U.S. Census, 2006). By the end of 2009, nearly 99 percent of public libraries reported having Internet access as a service to their clients (Bertot & McClure, 2009). During this period, the number of Web pages had grown exponentially; by 2000, Google reported an index of 500 million pages (Sullivan, 2005).

Vast amounts of content were added to the Web each year and search engines that facilitated access to Internet content became one of the major success stories of Web 1.0. The huge amount of content combined with powerful, free search engines changed how people used computers for information seeking. During this time, searching became a dominant activity of the Web (Rainie, 2005). Internet search engines not only provided viable alternatives to traditional sources of information, but they quickly changed user expectations regarding the information provision role of libraries. Graphical browsers, such as Netscape and IE, allowed users to access content through browsing or to facilitate searching. In many ways, searching represents both a defining characteristic and a technology of Web 1.0; the “pull” model for information access, where information is actively sought and requested, is a dominant theme of Web 1.0. Users formulated and submitted queries, resulting in lists of Web pages in the form of static HTML files, which could then be retrieved and displayed. To the majority of users, Google exemplifies Internet search. Google is both the pinnacle of Web 1.0 technology and a model of Web 2.0 services, highlighting the sometimes-blurred boundaries between Web 1.0 and 2.0.

WEB 2.0

Early references to a second generation Web appeared in the late 1990s in the context of the Semantic Web; a Web that would be based on XML such as that described by Bosak and Bray (1999). The Semantic Web, envisioned as a future “intelligent” Web, is now referred to as Web 3.0 in part because Web 2.0 has come to have a different meaning. Some views on Web 3.0 are discussed later in this chapter. There are various definitions and descriptions of Web 2.0 as well; it seems to mean different things to different people in different contexts. Anderson and Wolff define it simply as “web 1.0 that works” (p. 125). Most definitions describe it as the collaborative, social, interactive, blogging, twittering, flickring, widget utilizing, mashup-driven Web that is collectively referred to as Web 2.0. In this view, the central idea of Web 2.0 is about delivering Web applications. Tim O'Reilly (2005), credited with popularizing the term in 2003, described it as:

Web 2.0 is the network as platform, spanning all connected devices; Web 2.0 applications are those that make the most of the intrinsic advantages of that platform: delivering software as a continually-updated service that gets better the more people use it, consuming and remixing data from multiple sources, including individual users, while providing their own data and services in a form that allows remixing by others, creating network effects through an “architecture of participation,” and going beyond the page metaphor of Web 1.0 to deliver rich user experiences. (para. 1)

Allen (2008) argues Web 2.0 is really a conceptual frame with technological, economic, philosophical, and user components. He observes that:

Web 2.0 is a shorthand term for many different things, some in conflict, some overlapping but marked especially by the fact that they are ontologically non-compatible. In short Web 2.0 is about ideas, behaviours, technologies and ideals all at the same time. Moreover, its distinctive assertion of a change in state, from Web 1.0 (a term that was never used in any case) to Web 2.0, begs the question of the degree to which this change has actually occurred or may be occurring *because* of something new, or simply involves a re-expression of things previously understood as “the Web,” but placed in a new arrangement or seen in a new light. (para. 4)

Lankes (2011) suggests that Web 2.0 is not really about blogs and wikis, or even about user-centered design, but about making users (who he prefers to call “members”) themselves participants in a process that focuses on “where the user, or now member, fits into the system, and consequently, how that system handles the member’s language” (p. 38). In his framing, these “participatory” or “conversational” technologies should allow members to build their own customized system by combining small applets of limited functionality to suit their needs. He concludes this approach will be a common feature of the future Web, however it might be numbered or named.

In the context of libraries, Web 2.0 thinking embodies all these facets—it is a technology, a philosophy, a business plan, a behavior, and a participatory model to engage users. This view of the Web led Michael Casey to coin the term “Library 2.0” and launch an associated blog in 2005 (Casey & Savastinuk, 2007). Again, the 2.0 brand applied to libraries is generally described rather than formally defined. Casey and Savastinuk suggest Library 2.0 is:

- A model for constant and purposeful change.
- Empowers library users through participatory, user-driven services.
- Through the implementation of the first two elements, Library 2.0 seeks to improve services to current library users while also reaching out to potential library users. (p. 5)

Note that technology is not an explicit element of their model. This is deliberate on their part—they acknowledge technology has an important role to play in developing services, but they emphasize that it is just a tool rather than an end in itself. However, the technologies of the Web have been the primary focus of this text. Examples of these Web 2.0 tools and their related technologies were considered in Chapter 2. Much of Web 2.0 is viewed as part of the social media phenomenon, which is revisited later in this chapter in the context of their use in libraries.

WEB 3.0

With the emergence of Web 2.0, it is naturally predicted that Web 3.0 is just around the next virtual corner; in fact, many assert Web 3.0 is already here.

As with Web 2.0, views differ on the nature and definition of Web 3.0, or even if discussions of the future of the Web should be framed in this numerically sequenced way. One view of Web 3.0 describes it as primarily a virtual reality-based Web (Bell et al., 2007). More often Web 3.0 definitions are aligned with the view of Web founder Tim Berners-Lee, who describes it in terms of a semantic Web (Shannon, 2006). Other descriptions of that view refer to Web 3.0 as the worldwide database; still others anticipate a data-driven semantic Web combined with “smart agent” technologies will be the path to a future Web that will become a form of artificial intelligence.

Virtual Worlds

During the last decade, the immersive Web of virtual reality worlds was often touted as the next generation of the Web (Alpcan et al., 2007; Bell et al., 2007). The use of virtual worlds has lost some earlier momentum, but they are considered here because many librarians were among the earlier explorers of these technologies and many believe these worlds are still relevant. Although there are hundreds of virtual world applications, Second Life (SL) is probably the best known and it will serve as a model to explore this view of the Web and its future.

SL is a multiuser virtual reality environment, referred to as a MUVE (Multiuser Virtual Environment). SL (www.secondlife.com) allows users to join without cost. Participants create an avatar, and attend a virtual orientation. SL has its own economy, based on “Linden dollars,” and people can earn and spend this currency through their virtual activities in SL. Avatars can be customized, have wardrobes, move by walking or flying, and most importantly, interact with other avatars. Interaction typically is through keyboard chat features, but voice chat communication is also possible. Many users get very involved with this virtual world, buying and selling real estate, or starting virtual business ventures. SL evolved from the MUD (Multi User Dungeon/Dimension) programs that also gave rise to various gaming environments such as World of Warcraft. Modern gaming worlds are known as MMORPGs (Massive, Multiplayer, Online Role-Playing Games). However, SL is not actually a game; a game has points, levels, or other goal-oriented activities generally lacking in SL. SL is just that—a virtual life that can be experienced and explored. There many such virtual worlds; as of the end of 2011, there were an estimated 1.7 billion registered users of about 400 virtual worlds (KZERO Worldwide, 2012). These reported total user populations are impressive, but possibly misleading because of the churn rate, which refers to those who try it once and do not return, perhaps because they disliked the “game” aspect of the software used to create or use virtual meeting space.

SL is described as “free,” but there are costs. Land ownership requires a membership with a monthly fee and Linden dollars are needed for various other activities in SL. These costs can also change—a few years ago, many users were affected when Linden Labs discontinued its educational discount. That discount was restored, but many educational users have not returned. Other costs are the time to download, install, and configure the software. The bandwidth required for this activity could also be problematic for organizational networks. SL is an uncontrolled environment, which raises potential liability issues for public entities such as schools and libraries. For example,

shortly after the shootings at Virginia Tech, an avatar visiting Ohio University's SL campus virtually fired at fellow avatars causing much distress among other users. Sexual harassment or cyberbullying could raise questions about requiring that class activities take place in SL (Bugeja, 2007). Virtual worlds also attract some who do not view harassing behaviors as inappropriate. Virtual worlds have a "griefer" subculture, a term from the late 1990s applied to antisocial behaviors in multiplayer games, where some users seek to disrupt the game, to drive new participants away, or just to be a nuisance in a way that their peers find amusing. This culture has found its way into SL, and there are notorious examples of their actions, such the disruption of a virtual CNET interview or the vandalizing of candidate John Edwards' virtual campaign headquarters (Dibbell, 2008). These issues notwithstanding, many libraries and universities began using SL as a way to reach virtual clients. Calongne and Hiles (2007) reported more than 100 universities had a presence or met classes in SL. The Alliance Library System began exploring SL for its 259 member libraries in 2006 and for a time, had a significant presence called the Info Archipelago (Bell, Pope, Peters, & Galik, 2007).

However, the predictions of the acceptance and use of virtual world have not been borne out; for example, the inflated 2007 Gartner prediction that by 2011, 80 percent of Internet users would be active within virtual worlds (Gartner says, 2007). This highlights the challenge of predicting how a technology will evolve, and clearly there are times when all the excitement and media attention surrounding something new can have both hype and substance. Many emerging technologies appear to experience what Gartner refers to as a "hype cycle" where initial enthusiasm wanes during a "trough of disillusionment" before becoming mainstream, and that could apply to virtual worlds (Gartner's hype cycle special report, 2009). There are still committed people engaged in serious efforts to explore SL to support professional activities and educational opportunities. A 2012 survey of a variety of colleges and universities found examples of academic librarians using SL for public services such as reference, Web development, collection development, and instruction (Cote et al., 2012). Even if SL as it exists does not become everything its promoters once suggested, other virtual environments specifically tailored to educational applications are likely to emerge.

The World Wide Computer

Another view of Web 3.0 relates more to how grid and cloud computing, mobile devices, and creative applications are resulting in a smarter Web. One of the disruptive aspects of Web 2.0 is the notion of the Web as a platform for delivering applications. Carr (2008a) and others describe how the World Wide Web is becoming the World Wide Computer, comparing it to an earlier disruptive technology: that of electric power at the turn of the twentieth century. This shift was alluded to in Chapter 2 in the section on cloud and grid computing. The parallel between the early days of electric power and computing is intriguing and compelling; companies are finding less need to build, support, and develop customized computing and data storage solutions if they can be accomplished more efficiently via the Web. Web technologies and services are

converging with shifting user attitudes to a tipping point in favor of this paradigm. The use of cloud and grid technologies for both data processing and data storage proves users are becoming comfortable with the idea that their content can be stored remotely, and the seamless integration of Web technologies, broadband access, and mobile devices into our computing life makes the distinction between local and remote locations insignificant. Many companies are approaching their computing needs in a similar fashion, and not just for storage. Cloud and grid computing, also called utility computing, utilizes Hardware/Software) as a Service (HaaS and SaaS) making it possible to rent VMs and data centers as a Web service. Companies such as Amazon have created a huge infrastructure to support its own business activities and they can sell some of this capacity as a service. Amazon CEO Jeff Bezos compares generating your own computing capacity to be as unnecessary now as it is to generate your own electricity (Reiss, 2008). The availability of cloud computing permits entrepreneurs to start a business without the capital required to build their own infrastructure. However, there is a tradeoff with cloud computing just as there is with electric power: If the grid fails, your systems go down with it, so some local backup systems are still needed.

Semantic Web

As introduced in Chapter 12, the semantic Web (aka the “Big-S Semantic Web”) initiative was formally launched by the W3C in 2001, where it was described as “a common framework that allows data to be shared and reused across application, enterprise, and community boundaries” (W3C, 2001). Web founder Tim Berners-Lee has long promoted the idea of the Semantic Web as the ultimate future Web, asserting that an XML data-centered Web will enable direct data exchange among applications and enhance intellectual access to content through search engines designed to exploit its structure (Berners-Lee, 1999; Berners-Lee, Hendler, & Lassila, 2001). Although this vision would utilize XML, it also depends on other technologies and standards. Standards for the Semantic Web include the RDF, which supports metadata exchange and interoperability, and OWL, designed to facilitate content processing by applications, documented at W3C. Berners-Lee is not alone in the view that this metadata-rich Web of interlinked data sources and applications is becoming the next generation Web.

The semantic Web is not just a possible future—it is important now as elements of that broader vision are being applied by content producers and utilized by search engines. Web 1.0 was about connecting documents across many hosts. Web 2.0 is about applications, often run within an extensible browser. The newly emerging semantic Web is about connecting data within those documents and applications. Metadata is a key component of a semantic Web, enabling programs to share and interpret Web data (Fay & Sauers, 2012). Librarians have long understood the importance and power of metadata for information access, and they are applying this understanding to this new context. They are developing new standards for metadata, replacing the previous Anglo-American Cataloging rules with the Resource Description and Access (RDA) standard that refines the data elements associated with bibliographic

materials to enable more granular connections. At the document level, the conceptual model of the Functional Requirements for Bibliographic Records (FRBR) maps relations among entities represented in catalogs, linking together various versions of a work. In addition to formal cataloging and classification, librarians are also enriching content with embedded metadata using RDFa and microdata formats. Metadata is also a key component of the OAIS model discussed in Chapter 11. Collectively, these activities and standards support new access options with semantic search.

Not Dead Yet

In 2010, a striking cover of *Wired* proclaimed, “The web is dead.” Their obituary listed the primary cause of death as a mix of smartphones, apps, IM, music and video services, and peer-to-peer file transfers. In his book *The Future of the Internet and How to Stop it*, Zittrain (2008) compares these “appliance” approaches to what he describes as the “generative” technologies of the earlier PC environment, expressing his fear that they might result in less innovation. The distinction between the Internet and the Web has been discussed throughout this text—the Internet can support many activities, but technically, the Web is about Web pages delivered via HTTP. The app-centered approach does raise concerns because of the control one company such as Apple can have on app development and distribution. In addition, some of the openness associated with the Internet is giving way to closed subscriber-based premium services. However, to paraphrase Mark Twain, it appears that the reports of the death of the Web are greatly exaggerated. It is true that many users are using apps and are willingly relinquishing openness in exchange for reliable tools that seamlessly integrate their online activities. However, they will still also need to access the open Web as viewed in a browser, and hopefully the site they need will be one that has been optimized to ensure it both looks good and works on their mobile device.

LIBRARY 2.0: WORKING IN THE WEB

Although this text has been focused on technologies, examples of how they have affected libraries have also been considered. This section highlights examples of the applications of these interrelated technologies in libraries as well as to the services they provide to their communities; these Web 2.0-enabled services are often described as giving rise to Library 2.0. The “library 2.0” name has lost some of its meaning and is somewhat limited or overstated (Tennant, 2011). As Lankes (2011) points out, libraries have undergone many transformations throughout several thousand years of history. However, the 2.0 label is still used in the literature and serves as a way to organize and discuss these topics, many of which overlap and interconnect. Web 2.0, the social Web, cloud computing, and the dramatic shift toward mobile are converging to reinforce each other and support library 2.0. Many of these tools are also interconnected—social networking sites connect to Twitter feeds, which can connect to text messaging services.

The Googlization of Information Seeking

At its inception, Google laid down an implicit challenge to libraries in its mission statement “to organize the world’s information and make it universally accessible and useful” (Google mission statement, 2008). This mission is very much the same as that of libraries, and the sense of competition has increased with the development of other Google services such as Google Books and Google Scholar. Studies of how people seek information show that the competition is not just an abstract concern but is very real. As early as 2002, a Pew survey found 73 percent of college students seeking information were using the Internet rather than a library; only 9 percent used a library more than the Internet for information needs (Jones & Madden, 2002). Another report found 89 percent of college students begin their searching with a search engine compared to just 2 percent who start with a library website (OCLC, 2006). That study also found 93 percent of these students report being satisfied or very satisfied with their Internet searches compared to 84 percent reporting the same level of satisfaction for librarian-assisted searches. Other studies confirm the shift from libraries to Internet search engines for the information seeking of many types of users. A British Library study concluded that university collections are not meeting the needs of the Google generation and could be “swept aside by history” unless they adapt and make closer links to Internet search engines (Gill, 2008, para. 1). Search continues to be one of the most popular uses of the Internet; in 2012, 73 percent of Americans report using search engines, and 91 percent of those users say they find the information they are seeking (Purcell, Brenner, & Rainie, 2012).

This shift in search behaviors and attitudes is occurring across many demographic groups. An illustrative example is from an interview on *NPR Morning Edition* concerning a doctor’s search for new treatments for a patient with a genetic disorder. His research suggested that a drug that blocks a protein called TGF-beta could help this condition. The narrative continued:

It generally takes years to develop a drug. But Dietz went to his computer, pulled up Google, and typed in “TGF beta-blocking drug.” Up popped references to a drug called Losartan, which was on the market as a blood pressure medication but also happened to be very good at stopping TGF-beta. Dietz says it would have been hard to design a better drug. (Kestenbaum, 2008, para. 11)

While anecdotal, it is informative that the doctor turned immediately to Google to find this information instead of a formal database such as PubMed. A preference for easily available sources such as peers is neither new nor unique (Coleman, Katz, & Menzel, 1957). Bates (2002) notes physicians often depend on sales representatives rather than the medical literature for new drug information. That such informal sources are generally preferred over more formal database sources for most queries is not surprising. Google, with its availability, convenience, ease of use with conversational search, and usually relevant results, has become almost like a trusted friend for many.

The public sees little distinction between search engines and libraries, especially because Google offers additional research products and services and is

engaged in large-scale digitization projects. However, where some see a threat, others see opportunity. Abram (2005) views the emergence of Google as healthy competition that will not only drive libraries in new directions but also make teaching roles even more important, especially in the context of information literacy. Jerilyn Veldof, director of the University of Minnesota Libraries undergraduate initiatives, also sees this as an opportunity, observing that “libraries are building a bridge between the Google paradigm these students are used to and a much more sophisticated research-library approach to information” (Coventry, 2006, para. 5).

Library Systems

Libraries must deploy and manage many different and often interconnected systems. These include the catalog and the many aggregators of full-text databases as well as services designed to support the increasing collections of digital magazines, audiobooks, and eBooks. These various systems have often existed in separate siloes that creates inefficiencies and additional effort for the information seeker. Next generation library service platforms (LSPs) seek to provide better integration of all the services available as well as providing better and more seamless discovery services. Several overviews of the current state of the library system ecosystem are by Enis (2014) and Breeding (2014).

The Integrated Library System

The Integrated Library System (ILS) has been a core library technology for decades and was available locally in libraries before the Internet was widely available. With the emergence of the Web, the ILS has become a mainstream online resource that affects every facet of library operations, from acquiring and cataloging materials to managing patron information and circulation. It determines many of the services that can be delivered via the library website through the OPAC. The details of implementing a large-scale automation project or migrating to a new system is well beyond the scope of this text, but as a central component of the library website that intersects with many Internet technologies, some discussion of it is warranted. A good source for more detail on selecting and implementing an ILS is from Webber and Peters (2010).

The ILS must support many functions, and it does so through various interrelated but separate modules, each of which can be customized as needed. It usually is just one part of a complex environment, and must interact with a number of other products such as SFX link resolver services, OCLC cataloging records, ContentDM for digital repositories, ILLiad for interlibrary loans, EZ-Proxy for remote access to licensed databases, as well as options for customized local code or third-party add-ons (Kelley et al., 2013). The circulation module handles inventory control, bar codes, item check-in/checkout, and the policies or loan rules that govern fines and overdue notices. The patron database has all patron information (name, address, patron ID, etc.), which can be linked to borrowing records. Acquisitions and approval plans must be managed, and materials cataloged either by creating or importing bibliographic and authority records from templates, or from outside sources with the Z39.50 protocol.

The ILS must also generate or customize needed reports. Finally, the public access catalog (PAC or OPAC) module is the public face to the collection and the primary discovery tool for library resources; it is often the main feature of the library website. The OPAC must handle authentication of users for access to user accounts and licensed electronic resources. The look and function of the OPAC can be extended by many add-ons, but it also has built-in limitations. One note about the name “OPAC”—the term is a familiar one in the library profession, but many would like to see it retired. Both the “online” and “public access” parts of the name are remnants of the time when these systems first came online and open to public use, but the term now seems anachronistic and worse, it is often meaningless to the public. For the rest of this discussion, the OPAC will be referred to simply as the catalog, and it is assumed to be both open to the public and online.

There are both proprietary and open-source options for ILS solutions; the choice between them depends on many factors, including budget, local expertise, and the longer term strategic plan for the library. Proprietary options include Voyager, Apollo, Millennium, Symphony, and Library Solution. Major open-source options include Koha, Evergreen, and OPALS. Costs can be significant for both—open-source options can be downloaded for free but have fees for support as well as needing local expertise. Within both the proprietary and open-source market, the ILS solution might be:

- A turnkey system: As the name implies, a “turnkey” system is purchased from a single vendor who provides all needed components and technical support.
- A stand-alone system: The software is purchased and installed locally on a single computer or a server in a LAN/WAN environment. The stand-alone option requires local management of a number of issues such as system updates and backups.
- A hosted system: Similar to a SaaS solution except you purchase the software that is then hosted on the vendor’s hardware and servers.
- A SaaS system: A cloud computing option where the hosted ILS software is not purchased but accessed through a subscription service (see Chapter 2 for more on cloud computing and SaaS).

There is some debate on what comprises the “next generation” ILS. Those migrating to new generation systems identify both internal and external benefits. Internally, there are hopes for cost savings and efficiencies through cloud solutions, better management of digital resources, or improved workflow with self-checkout or online renewal. The next generation ILS might also improve the public face of the catalog, for example, with a single search box and more relevant results expected by Digital Natives (Kelley et al., 2013). These changing expectations for this part of the ILS are explored further below.

The Library Catalog

The library catalog has both staff and public functions. It has traditionally been the public face of the library and the main tool used to access to its

collections. Debates about its future in a Web-based digital world are not new. A 2006 report prepared for the Library of Congress represents the side of the debate that suggests that users are routinely bypassing the catalog as a discovery tool and that it does not represent huge areas of useful content; the report pronounced that the catalog was “at the end of its useful life cycle” (Calhoun, 2006, p. 10). A rebuttal by Mann (2006) argued many assumptions in the report were flawed, and that a distinction should be made between academic scholarship supported by libraries and quick information seeking. While acknowledging that libraries might not be the first source users turn to, they are still important to research and regularly used by students and scholars. However, fair or not, users inevitably compare their experience with library catalogs to their experience with using search engines or alternatives such as Amazon or Google Books. Libraries must update and revitalize the library catalog interface and its feature set to better meet enhanced user expectations. In addition, to compete with the one-search Google world, effective federated search discovery tools are needed to better connect users to the high-quality information in the vast array of heterogeneous library databases.

The preference expressed by many users for Internet search engines over libraries is documented but the reasons for their preference is not always known. One important reason the libraries might be less used is that many users associate them exclusively with books. Items found in a catalog search also require additional effort to retrieve compared to the instant gratification of answers from a search engine. It is reasonable to suspect some of the preference has to do with dissatisfaction with the catalog experience when compared to using a search engine or to searching sites such as Amazon. In a study of academic library catalogs, Mi and Weng (2008) found that user behavior and expectations have changed, but many library systems have not adapted to these new expectations. They identified interface issues such as confusing field labeling, displays of results that did not retain original search terms for possible modification, and failure to provide availability information with search results. Often the catalog default search assumptions for term processing were not clear or required the use of explicit Boolean operators. In addition, they found a general lack of value-added elements users have come to expect such as text summaries, tables of contents, reviews, or links to vendors that provide those enhanced display services. Many library catalogs offer limited access to content, are hard to use, and lack features such as relevance ranking and connections to social media.

User and usability research suggests a need for a new generation of library catalogs that are better at connecting users to the wide array of information sources and at presenting the results in creative ways that meet user expectations. Coyle (2007) calls the Web 2.0 audience “User 2.0” and suggests new generation catalogs must go beyond 1.0 thinking to incorporate features enabling users to participate and add content and value to the experience. There are interesting models for the next generation catalog; for example, LibraryThing (<http://www.librarything.com/>) is an online community catalog of 84 million books created by about 1.7 million members with features such as tagging, shared recommendations, and blog connectivity (Zeitgeist overview, 2013). Many libraries are implementing participatory features and adding new content such as images of book covers, tables of content, abstracts, and reviews.

Changes in metadata standards will in turn improve the access provided by catalogs and the search options they offer. As this process goes forward, fundamental questions must be addressed about what the catalog should contain and how it connects to other information systems. Decoupling the catalog from the rest of the ILS is one approach. Another option is to replace or supplement the catalog with a new unified finding aid. Another alternative is to develop dedicated mobile apps for different catalog functions. Breeding (2007) identified a wish list of features for the next generation catalogs that includes:

- A single point of entry for all searching (federated search access to all materials) through a simple keyword search box;
- A more intuitive Web interface;
- Enriched content and user contributions, for example, tables of contents and user reviews;
- Relevancy ranking of results instead of just sorting by fields such as date;
- Spell checking and search suggestions;
- Recommendations and related materials.

Library catalogs could benefit from semantic Web approaches that apply fuzzy Boolean to searches—far too many searches return zero results. They could also seek to go beyond spell check to provide semantically relevant auto-complete search phrase suggestions as Google does when you type in a search, or mimic the shopping cart of Amazon. For example, my university's library has a book delivery service that can deliver a book within a day to a nearby campus location. To request this service, the patron must login with their university credentials and then choose the location for the delivered item. However, for multiple items, the delivery location must be identified again for each request during that same session, even though the user has already logged in and provided that information once. There is no "shopping cart" to save items as you search for other items and no single "check out" option to request delivery of all items to the desired location when the session is finished. There is no personal profile feature that allows a default delivery preference to be set. Compared to the Amazon experience, the repeated call for the delivery location in the catalog seems unnecessarily redundant.

Public library users seeking eBooks are frequently frustrated by the need to establish multiple accounts for various services. In addition to a library account, a user might also need separate accounts for OverDrive, OneClick Digital, Zinio, and Adobe Digital Editions to access eBooks, audiobooks, or digital magazines. Another source of frustration are catalog features that do not work across all browsers.

Some of these issues are caused by the way various features are enabled in the local configuration options and others are imposed by the publishers that own and license the content, but many limitations are imposed by the vendor's software design. Librarians have long been held captive by a limited set of programs capable of managing the many functions of an ILS. This is changing with the development of open-source options such as Scriblie (<http://about.scriblie.net/>), Koha (<http://www.koha.org/>), and Evergreen

(<http://open-ils.org>) that offer catalog functionality with 2.0 features. Weber (2006) describes one library's experience in making a transition to the Evergreen open-source catalog. Coombs and Hollister (2010) identify many other open-source applications of interest to libraries.

Open URL and Link Resolvers

One of the issues that complicates the delivery of some content through the catalog is that it resides in other locations that restrict access to licensed users. Link resolvers were introduced in Chapter 7 as a way to associate a name alias with a URL that performs a redirect to the site, such as with Google's URL shortener (<http://goo.gl>) or TinyURL (<http://tinyurl.com/>). This technology has been adapted by libraries to make OpenURL links from catalogs to a source such as an online article. The OpenURL carries information about the library's link server address that is then appended with a query string containing the necessary bibliographic data. In addition, these services can perform rights management and authentication to ensure access is limited to appropriate users. The SFX software, developed in the late 1990s, was the first OpenURL link resolver service for libraries. It was purchased by Ex Libris in 2000 and it is used by more than 1,500 libraries (SFX, 2012). The Library of Congress lists a number of other OpenURL vendors at www.loc.gov/catdir/lcpaig/openurl.html.

Discovery Tools

As discussed earlier, many of the library's content sources are located in separate silos that must be searched separately because catalogs frequently lack effective metasearch (aka federated search) options. This lack of integration creates added effort for the consumer and can reduce their awareness of what is even available through their library. Commenting on the results of the Pew Research Library Series study, Price observes that "these reports also show that a lot of people have no idea what the library offers" (as cited in Chant, 2014). This could in part simply be a PR problem, but poor discover tools surely exacerbate the issue. The "Googlization" of searching has given rise to users who are less patient with the complex options that libraries offer and who prefer Google's simple "one box" search interface. The cliché that "only librarians like to 'search,' everyone else just wants to find" has more than a kernel of truth. Libraries must continue to develop discovery tools that provide a seamless user search experience across the large number of separate databases they manage. However, achieving this goal is sometimes a more complicated and messier process than simply designing better search technologies and simplifying the interface. As in the case of eBooks where publishers control much of the process, effective metasearch is not just up to libraries—it also requires the cooperation of competing content owners and aggregators who have different priorities and interests to protect.

A related issue is what is being delivered by the search. Catalogs traditionally have delivered information about a container, but often what is sought is something inside that container—the chapter rather than the book or the article,

not the journal. A recent study that asked undergraduate students to compare federated search options to Google reported both were useful, but they preferred federated search for assignments that required research-related searching (Georgas, 2013). Federated search discovery system examples include the Encore discovery solution that provides single-search access to eBooks, articles, local collections, and books (<http://encoreforlibraries.com/overview/>) and the Summon service (<http://www.serialssolutions.com/en/services/summon>).

Tagging

One common thread for many Web 2.0 technologies that is especially relevant to libraries is the potential for user-generated tags for content description. This empowers users, and they seem to embrace these capabilities to create descriptive access points that work for them; 28 percent of Internet users report tagging activities used to categorize online content such as photos, news stories, or blog posts (Rainie, 2007). Tagging done with an unrestricted term vocabulary results in the creation of a *folksonomy*, a term that is a hybrid of *folk* and *taxonomy* (Peters & Bell, 2008). Folksonomies are unrestricted user-generated vocabularies that enable a “group think” approach to describing content. This process can result in a large set of assigned terms, which are often represented as a tag “cloud.” The cloud visualization can indicate term frequency by the size of words in the cloud; the larger the size of the word, the more commonly it was assigned by users to describe the content, as shown Figure 14.13 earlier in this text. Users often have a number of options for providing content description; in addition to tagging with words or phrases, they may post full reviews or generate numeric ratings. Such reviews have proved to be very popular within many consumer sites; the ability to hear about actual user experiences with a product has become an equally useful source of information to traditional review sources.

However, given librarians’ long history of creating controlled forms of content description many have understandably mixed feelings about fully embracing user-generated metadata. It is one thing to let people tag their own photographs, but it is something else to permit users to add descriptive tags to traditional catalogs. What does this mean for all the high-quality metadata that has been created through years of effort? Would users overwhelm record displays with tags that add little descriptive value? It is true that some user tags are of limited value, but it is likely that the expansion of descriptors used could be useful by permitting terms that users prefer, especially about new topics. Data mining techniques can be applied to the collective tagging of users to identify new terms and provide new sources of metadata. Even though librarians justifiably defend formal metadata as a means to enhance access to resources, there is evidence that alternate approaches can provide as good or better search results. Hawking and Zobel (2007) examined queries of a university Website utilizing subject metadata and they found metadata-enabled searching actually underperformed when compared to queries using anchor text. One reassuring aspect for librarians is that user tagging and traditional controlled vocabularies are not mutually exclusive strategies; allowing user tagging in a catalog does not mean that other forms of value-added metadata

must be abandoned. However, it does force librarians to reconceptualize information systems and view the world more from the “user 2.0” perspective.

The Library as Place

Over the last several decades, technology has dramatically changed how people use libraries, which in turn has shaped the design and utilization of library spaces. Card catalogs that once took up significant floor space gave way to computer terminals and printers. As libraries moved increasingly to digital resources and Web services, less space had to be devoted to print collections and more emphasis was put on creating spaces for community activities and collaborative learning. The idea that libraries are involved with information creation and dissemination is not new; many libraries create local repositories on many topics such as local history or genealogy. They also manage learning spaces such as writing and media centers, and provide instruction to the communities that use them. This learning center role of libraries is expanding to support the Maker culture movement that seeks to create information through various hands-on activities. Torrone (2011) suggests that public libraries could be retooled to become “factories” of learning with “hackerspacers” (a place for experimenting with electronics and circuitry), “Fab Labs” (places to create products), and “TechShops” that combine both these types of activities. Some of these are made possible by technological advances in 3D and cutout printing, inexpensive circuit kits, and devices such as the Raspberry Pi (a simple and customizable computer). However, Maker spaces are not just about high-tech; they can also be developed to support any learning activity that needs to be hands on, from woodworking to knitting. Some of these ideas may sound familiar to the generation who grew up with Popular Electronics magazine and Heathkits, but their rebirth in the library context offers interesting ways to engage new communities. As Mitchell (2014) observes, the core mission of libraries is about information flow and access, not just about books, and library Maker Spaces could support new forms of collaborative information creation and dispersal.

Social Media

The collaborative nature of Web 2.0 is largely due to the rapid development and acceptance of various forms of social media. A 2013 Pew report estimated that 72 percent of U.S. online adults are using social media (Brenner & Smith, 2013). The power of social media is demonstrated not just by the huge number of personal connections it fosters but also by its larger role in affecting societal change. On the lighter side, it has made us participants in entertainment and reviewers of products, but it has also had a serious role in bringing the attention of the world to protest movements and disasters as they happen. As noted in Chapter 2, many of the tools of social media are associated with Web 2.0, and the two terms are often used interchangeably. However, social media is not just about the tools—it is about building social capital. Its success is driven by willing participants who create and monitor user-generated content (UGC). One definition of social media is from Kaplan and Haenlein, who describe it as

"a group of Internet-based applications that build on the ideological and technological foundations of Web 2.0, and that allow the creation and exchange of user-generated content" (2010, p. 61).

Social media were originally designed for individual personal use, but they now are part of the strategic plan for many institutions and businesses. A poll of cooperative marketing executives found they expect social media expenditures will rise from 8.4 percent to over 20 percent of their marketing budgets during the next five years; they see it as critical to making existing practices better (Hempel et al., 2013). The same mandate applies to libraries; it can be used for outreach and facilitating participation and engagement, but as in the business model, the overall goal should be to improve current practices. When the Web first came on the scene, librarians were early adopters of that technology as a way to reach their communities, and adoption of social media has been equally rapid. However, just as some underestimated the time and effort needed to build and maintain a website, this can also be true for social media use. In the early Web, the goal was often just to "get out there" by putting up a presence and then not paying much attention to it. As time went on, libraries quickly recognized a successful website required a deeper understanding of user needs, a clear vision for its mission, and personnel responsible to keep it current and to collect analytics and feedback for continuous improvement. Social media requires a similar ongoing commitment; successful use involves more than just creating a Facebook presence and waiting to reap the benefits. Another parallel of social media with earlier Internet technologies was the need for library policies related to their use. A social media policy should encourage good judgment in its use and anticipate possible problems regarding confidential information, copyright, the expression of personal or controversial opinions by employees, and friending guidelines. There are monitoring software tools that can provide data about the effectiveness of social media use and guide decisions on their implementation.

Building social capital requires tailoring the tools and the messages to the audience and for libraries, there are multiple demographics to serve. To determine the best social media strategies, some type of needs analysis process is necessary, similar to that employed in Web design, to articulate the goals that are to be achieved and the means to accomplish them. Understanding where the people you want to connect with are in the social Web and what they do there is essential. To reach them effectively, multiple tools are often employed. For a time, younger audiences preferred Facebook to MySpace, but it now reaches many older adults; the number of users 55 and older grew by 80 percent in the three-year period ending in 2013 (Saul, 2014). Some activities might be better suited to Pinterest or Twitter, depending on the intended audience and the nature of the information (Madden et al., 2013). Tumblr is most popular with the 18–29 age groups, and unlike Pinterest, men and women use it equally (Desilver, 2013). However, these preferences are a moving target—recent evidence suggests that as Facebook has become a more mainstream SNS, many teens see it as "uncool" and are abandoning it in favor of other platforms such as Twitter or Snapchat (Pressman, 2014). Solomon (2013) discusses the importance of finding your library's voice, and emphasizes that a formal tone and institutional jargon is off-putting and not effective communication in social media. She also gives great examples of how to humanize the formal language used in typical posts librarians need to make on social media.

Social Networking Sites (SNS)

Social media and the common social networking sites were introduced in Chapter 2, where they were described as the descendants of earlier virtual communities such as Geocities. Facebook, MySpace, Foursquare, LinkedIn, Google Hangouts, Tumblr, Pinterest, Instagram, and Twitter are all well-known examples and are ranked among the top sites (Top 15 most popular SNS, 2013). However, any site that depends on broad communities of engaged participants, such as Wikipedia, is also a type of social media. To leverage social media successfully, libraries need a formal plan that includes a multifaceted approach to social media in order to reach the sometimes-differing communities each site attracts. Facebook is the dominant player, having more than a billion active Facebook users, but MySpace, which was once the largest such site, relaunched itself with a redesigned site, mobile app, and a music streaming service (Hampton, 2011; Tam, 2013). Pinterest has attracted many users who use it to collect, organize, and share their interest by “pinning” content to a board. Evernote is similar in allowing users to organize links and content, but uses notebooks instead of boards.

Social media is a force that should not be ignored; between 2008 and 2011, the number of those using social media doubled and the demographics of its audience are shifting to include larger numbers of older adults. The social media demographic divide appears to be narrowing; Pew research reports that between 2009 and 2010, adults over the age of 50 were joining social networking sites at a higher rate than that of the 18–29 age group (Madden, 2010). Libraries can serve these groups by adding sessions on how to use social media as part of the IT training that has been traditionally offered to midlife adults and seniors. Schull (2013) provides many examples of such initiatives such as Portland’s “Cyber Seniors” program and Hennepin County’s “Senior Tech Day.” Libraries are following many other businesses in formalizing the role of social media in their business plan. These popular sites present an opportunity to connect with new users on their turf. Rapacki (2007) describes using a MySpace presence at the Wadsworth (Ohio) Public Library teen department as a means to engage teens in the community. The Hennepin County (MN) public library (<http://www.myspace.com/hennepincountylibrary>) uses MySpace for a similar purpose. The New York Public Library (NYPL) reports a Twitter following of 250,000 members and featured a 30-day celebrity quote campaign that boosted library card sign-ups by 35 percent in a month (Dankowski, 2013). Other public libraries have found social capital extremely valuable when facing budget cuts as a way to engage the community in a conversation about what the library means to them and gain additional support.

Facebook is commonly used by university libraries to connect with students. Pinterest is being used to support library reading programs and reading lists, share new acquisitions, and display historic archives. Social media has many internal professional applications for advancing career opportunities, from acting as an online rolodex to making new connections. LinkedIn (www.linkedin.com), with its focus on a more professional and business audience, has become a necessary networking tool to support the career goals of many. Table 16.2 lists the major social sites; various sources report different numbers for each site, but these estimates provide some comparative information about them. A simple way to explain how these sites are different is shown in Table 16.3.

TABLE 16.2 Popular Social Media Sites and Estimated Membership

Site	Estimated Monthly Visitors or Users	Description
Facebook	850 million–1 billion	The current leader in social networking. Users can create profiles, connect with friends, post photos and commentary, and vote to “like” a person, business, or service.
MySpace	70.5 million	An initial leader that has been relaunched with a music streaming service; many features similar to Facebook.
LinkedIn	110 million	A more professionally focused network; users create a profile, but it is less personal and more business oriented.
Twitter	250–550 million	The influential microblogging site. Users can post personal updates of up to 140 characters, follow other Twitter users, and tag content.
Google+	65 million	Google’s entry in social networking. Similar features to Facebook, but better promotes the creation of various social “circles” that can be centered around family, work, or friends. The “Huddle” is its group phone chat feature, and Hangouts are used to indicate you are available for chat.
Instagram	80 million	Primarily a photo sharing service with some manipulation tools. It has connections to the other main SNS (Facebook, Twitter, Tumblr). Users can also follow others and comment on their photos.
Pinterest	85 million	Based on sharing image bookmarks, users create pinboards on which they can post images, videos, and links of interest, called pins.
Tumblr	216 million	Acquired by Yahoo, Tumblr is a blogging platform that allows users to easily share photos, videos, and messages. Pinboards can represent different categories of interest and can be private or shared.
Four-square	33 million	Combines geolocation data to allow users to check in to share their location and see where their friends or a business of interest is in relation to it.

<http://expandedramblings.com/index.php/resource-how-many-people-use-the-top-social-media/>

TABLE 16.3 Social Media in a Nutshell

Site	Explanation in a nutshell
Twitter	I am drinking coffee now.
Facebook	I like coffee.
Linkedin	I am an experienced coffee-drinker and have coffee-related skills.
Youtube	Watch me drink coffee.
Instagram	Here is a photo of where I drink coffee.
Pinterest	Here are some good coffee recipes.
Foursquare	This is where I am drinking coffee

Instant Messaging (IM) and Twitter

These applications fall under multiple umbrellas—they are considered Web 2 tools that are important to both social media and the mobile Web. IM and chat applications have been around for some time. IM is less formal and immediate than email and has much of the appeal of text messaging. America Online (AOL) offered an early IM client as a service to customers, which is now branded simply AIM. Yahoo IM and Windows Messenger are other IM options. IM has been used in libraries for reference chat service for almost as long as email has, and clients such as Trillian allow chat to take place across competing services (Ciocco & Huff, 2007).

Twitter is described as a form of microblogging that allows short messages to be sent via the Web, IM, or mobile phone. Although a relatively new addition to the social Web, Murthy (2013) contextualizes today's Twitter by tracing its roots to other message services such as the telegraph, radio, and an intriguing message board system called the "Notifier" found in rail stations of the 1930s where users could post short messages on a strip of paper to inform others of their whereabouts. Twitter initially described its service as a way to notify those who care about all the little things in your life as they occur. These short messages are the virtual equivalent to the quick, casual "what are you up to" conversation you might have with friends you briefly encounter in real life. Although Twitter began primarily as a medium for updating friends and family and is still used this way, it has evolved from simple personal updates among friends to being used by both traditional news outlets and newly empowered citizen-journalists as an important source of updates on breaking news. It has proven its value to disseminating information when disaster strikes, such as the "miracle on the Hudson" crash landing and the 2011 Tohoku earthquake and tsunami. Twitter is not only used in reporting events, but has also been instrumental in shaping them, as in the Arab spring uprisings. Much of the success of Twitter in this context is due to its alignment with the mobile Web—any newsworthy event is now observed by many with smartphones that can post updates, photographs, and video in real time. Twitter account users are identified with the "@" sign and the messages (*tweets*) sent are limited to 140 characters. Messages can be categorized with *hashtags*, a descriptive word

preceded by a pound sign (#). Tweets can be retweeted by others, extending the reach of the original message to new audiences who may not follow the person who posted the original.

Twitter can be used to provide brief informational services in the form of online reference as do email and chat sessions or to create interest in an event or service. Some of these various messaging functions are being integrated into single client solutions, and Twitter is integrated into many other tools of the social Web. Microsoft has introduced Office Communicator, a client program to integrate messaging functions within Office and Outlook. It promises to integrate messaging with your contact list, allow rich IM content from Office applications, support PC-to-PC phone connections, and enable multiparty conversations with IM, phone, or video conferencing.

RSS Blogs and Podcasts

As described in Chapter 2, the blog, or the personal Web log, is made possible by the XML-based RSS protocol. They are essentially a personalized, interactive Web page that is easily updated and syndicated. Although most blogs are just personal journals, they have also become a successful mainstream information outlet. A blog is essentially the sharing of one person's worldview. Many allow for and encourage varying degrees of participation by readers who wish to post comments. Blogs have become increasingly important to information professionals as both a source of information and a vehicle to share information within a group.

Blogs have found their way into libraries in a number of ways. Blogs are replacing discussion lists as the main venue for professional conversations, whether for professional development, communication with user groups, or as a news outlet. Blogs are quite versatile, allowing embedded elements such as chat, audio, video, and slideshows that are pulled from Flickr or YouTube. Blogs with such embedded content are being used in place of a traditional website. Stephens (2007) identifies a number of value-added blogging activities for libraries, showcasing their use for readers' advisories, internal communication, and building interactive community-based websites. RSS features can be added to catalogs so users can syndicate their search and receive notification when new items are added that match the search criteria.

Podcasts are audio blogs. The term is from the words iPod and broadcast. Sometimes it applies to any downloadable MP3 file that can be put on an iPod or other MP3 player, but it generally refers to audio that can be syndicated and subscribed to, as with blogs. Podcasts not only allow anyone to create their own personal radio show, but they also have found many educational uses; a podcast is a possible way to deliver a lecture or audio program. The applications of this technology in libraries are numerous and include orientation tours, instructional audio on database searching, book or journal reviews, new book information, and events promotion (Jowitt, 2008). There are open-source tools for podcast creation such as the Audacity (audacity.com) audio recording and editing software.

Wikis

Wikis are another collaborative tool for Library 2.0. There are public websites that support wikis or they can be implemented locally. Probably the

best-known example of the power of crowd sourcing to build an information resource is Wikipedia (wikipedia.com), a collective encyclopedia built with the knowledge and expertise of the user community. Wikipedia has become a huge success—Nielsen Online reports that traffic to Wikipedia had grown nearly 8,000 percent in the five year period between 2003 and 2008 (Nielsen Company, 2008). Because of the wide subject coverage of Wikipedia, much of this traffic is due to referrals from search engines; Wikipedia commonly appears as one of the first links in any topic search. As discussed in Chapter 15, Wikipedia is also a common source for data provided by semantic search engines. This use of Wikipedia as a primary reference source is controversial because there is no assurance that all the information contributed by users is always accurate. In many ways, Wikipedia represents the crux of the broader dilemma faced by information professionals with many forms of user-generated content enabled by Web 2.0, that is, the concern that the resulting resources may not be authoritative or appropriate. There are legitimate concerns about the lack of traditional gatekeepers to validate and control Wikipedia entries, and there are examples of content manipulation to promote a biased agenda, such as the evidence of corporate tinkering with entries for public relations purposes (Hafner, 2007). However, Wikipedia also reflects the tremendous power and self-correcting mechanisms of user-driven content systems. Errors are usually quickly detected, and studies comparing its content in specific disciplines with a traditional source have found it to be quite accurate.

Aside from the controversy sometimes surrounding the use of Wikipedia as a reference source, wikis are like blogs in that they present opportunities to build and share content in many forums, and their use is growing. Wikis are being used in libraries to build in-house knowledge bases of shared best practices, to archive frequently asked reference questions and answers, to share committee and meeting reports, to create specialized subject pages, to support course content, and for conference programming (Fichter, 2006). As with a blog, wikis can also engage the user community and record their input. Library applications of wikis as well as blogs could be to involve users in library planning or to build content resources in areas such as local history and genealogy. Such community involvement not only engages library users, but taps into a huge pool of local experts to help develop a library resource.

Sharing Photos, Videos, and Bookmarks

Although Facebook has become the de facto photo-sharing site for many of its users, there are advantages to more specialized photo-sharing sites such as Flickr (www.flickr.com). Flickr accounts are available for free with a Yahoo email account and they offer a full terabyte of storage. It has tools that make it easy to upload, edit, tag, find, and share photographs with a private or public audience, depending on user preference settings. Flickr accounts can be used as a promotional tool or as a way to involve library users in a collective project. Flickr has many educational uses; an Educause report describes an architecture class in which students went out into a city on a digital-photo scavenger hunt to gather images of architectural styles; the images were uploaded, tagged, and incorporated into the course (EDUCAUSE, 2008). Stephens (2007) describes some best

practices for Flickr and libraries, including tailoring the profile appropriately for the intended use, tagging and organizing images, engaging the user community, and displaying the images on the website via RSS feeds. Flickr APIs are available for various Web mashups or mobile app development.

YouTube (www.youtube.com), now owned by Google, is similar to Flickr but it is for videos instead of photographs. It has become immensely popular, and librarians have used it for promotion and community engagement. YouTube is being used extensively by educators, who often create their own channel or embed other videos of interest into course materials. Because YouTube allows users to post commercial television and movies in addition to personal videos, it has raised copyright issues and challenges.

Shared bookmarking was developed to address both the limitations of browsers to manage large collections of bookmarks and to make them available when working at a remote location. The Delicious site (formerly del.icio.us), which began in 2003, was one early service. Google also now offers shared bookmarking with its accounts. For anyone who has a large set of personal bookmarks and who wants to be able to access them from anywhere, this site was a boon. Bookmarking was a feature introduced with the first browsers to help users keep track of, and return to, sites they discovered. However, personal bookmarks may quickly grow into a huge set of links, and the tools for organizing and managing them in a browser are limited. The bookmark itself has little descriptive information other than the title of the page, which is sometimes not provided, so finding a desired bookmark is often a challenge. In addition, users with computers in different locations find they do not have access to all their collective bookmarks. However, with a free account on Delicious or with Google, bookmarks can be set or imported from different computers to be organized and tagged with user-generated terms.

Librarians were quickly drawn to using shared bookmarks as an alternative to creating Web page subject directories that were a common feature on many library websites. Creating and maintaining subject pages was labor intensive. Rethlefsen (2007) describes how shared library bookmarks streamlined the process and empowered librarians to quickly post and tag interesting Web links they discovered. Flexible tagging provides a rich natural language vocabulary and can make it more likely users will find a link on a topic. “Task tagging,” a way to bundle links together for a specific audience or purpose, can facilitate better access to those resources. Although shared bookmarks are still useful for both individual and collaborative work, many libraries have turned to alternate strategies for creating subject guides such as LibGuides, discussed in Chapter 11.

Mashups and APIs

The term *mashup* comes from disk jockeys and musicians who create new compositions by “mashing” together other songs (Purvis, Sambells, & Turner, 2006). Fichner describes them as “a Web application that uses content from more than one source to create a new service in a single graphical interface” (Engard, 2009, p. 3). The proliferation of mashups has been driven by their usefulness and the willing of Internet companies such as Amazon, Google, and Yahoo to open up their data without licensing restrictions. Mashups often

412 Internet Technologies and Information Services

depend on a shared API to add functionality to your site. A common example of a mashup is the extensive use of the Google Maps API to overlay a new application on a site. Many sites have incorporated this functionality; for example, the Zillow real estate service (www.zillow.com) allows searching for homes for sale in a neighborhood or city and then displays them on a Google map with stickpins. There are many possibilities, for instance, restaurant or consumer review sites could display the physical locations associated with the review. Libraries could do a similar mashup connecting OCLC WorldCat with mapping to show locations of nearby libraries holding an item of interest. Google Map applications can be created with PHP and AJAX programming, but you can discover how to create maps without programming on the wayfaring site (www.wayfaring.com), which allows users to create and save customized maps they can link to later (Purvis et al., 2006). Other examples include connecting the catalog to a Google Books preview, connecting a map to archived photographs of that area, or mashing up shared bookmarks on Delicious into a subject guide. Steven Abrams's "APIs and libraries" (2009) and Michel's "Web service APIs" (2013) are good resources on APIs that are particularly useful for librarians. Mashups can be created without a shared API by utilizing the URL language employed by a site for data queries or through a RSS feed it distributes, but only after determining that such uses of the other site's data are permissible. Mashups can add value to the data delivered by a library website or catalog, and they are often not difficult to implement. Another good overview on creating mashups with examples for libraries is by Engard (2009).

MOBILE TECHNOLOGIES AND CLOUD SERVICES

Mobile devices and cloud-based services are taking over the computing world. The mobile device has become the center of all information activities for many people, and they expect information to come to them, rather than them going to it. There are many strategies to engage mobile users effectively, from mobilizing the website to creating specialized apps. There are a number of good sources on using mobile technologies and building mobile applications (Ally & Needham, 2012; Clark, 2012; Griffrey, 2010; La Counte, 2012).

Mobile Technologies

Mobile technologies present opportunities for rethinking how all information services, including the catalog, can be offered. As mentioned earlier, many library users might prefer dedicated apps specially tailored to specific tasks. Some vendors offer mobile add-ons for their catalogs and some libraries have built their own. However, building native apps from the ground up takes expertise and time, and there are third-party products available that can be customized to fit local needs. Boopsie for libraries (www.boopsie.com) provides mobile apps that have many features such as catalog search, ILS account information, GPS-aware location, text messaging to reference, and integration with many other information systems such as ESBCOHost. Options can include one-click access to OverDrive eBooks, scanning ISBNs to search, book checkout and renewal, and user reviews and recommendations. Such apps serve not just

remote users, but those in the library itself. Another advantage to the mobile users is that such apps combine the access function with the way they will use the resource; for example, by putting the eBook on the device where it will be read. Other useful database-specific library apps include WorldCat Mobile and PubMed Mobile Abstracts.

SMS

Short Message Service (SMS) is synonymous with text messaging. Although the idea of text messaging goes back to the 1980s, its rapid rise during the last decade has paralleled the shift to mobile cellular services. For the three-year period beginning in 2007 and ending in 2010, the number of text messages grew threefold to an estimated 6.1 trillion messages, or about 200,000 per second (SMS triples in three years, 2011). There appears to be a clear preference among Millennials for texts instead of live calls or email. Such data provide compelling reasons for libraries to explore using SMS to reach these users. Griffey (2012) describes their use for reference service and for various patron alerts or notifications.

For reference services that may require more back and forth exchanges, there are useful tie-ins between text messaging and various other types of IM or widgets that act as a bridge between a computer and the mobile device. The Meebo service has been acquired by Google, but there are alternatives such as imo.im (<http://www.makeuseof.com/tag/5-alternatives-to-meebo-for-web-based-multi-protocol-instant-messaging/>).

QR Codes

QR (Quick Response) codes are a new type of 2D matrix bar code that can be scanned with a mobile device camera to bring up additional linked information,

make a call, initiate a text message, or activate other phone apps. These are useful as part of a mobile/social Web-marketing plan or to link specific items or areas to more information, for example, to bring up an instructional video or an item record. QR codes can be easily generated with various free generators such as Kaywa (<http://qr-code.kaywa.com/>). They can be combined with other mobile-enabled features, such as connecting to a geolocation mashup. Figure 16.1 shows an example of a catalog QR code that is associated with the first edition of this text.



Figure 16.1 An example of a QR code.

TABLE 16.4 Useful Cloud Tools

Useful Cloud Tools	Description
Doodle polls (www.doodle.com)	A free scheduling tool that can be used to create polls to facilitate event planning and meetings.
Mail chimp (www.mailchimp.com)	A mail service that can manage email lists and specific messages for mass mailings or targeted subsets of the list. Drag and drop editor and templates provided to create elaborate newsletter-like emails. Delivery options include auto response and timed message release. Free and fee-based options.
SurveyMonkey (surveymonkey.com)	A free survey site with for creating surveys and analyzing the results. Premium fee-based service for added features.
Jing (www.techsmith.com/jing.html)	Free software from Techsmith for creating screen capture movies and sharing them on screencast.com
Eyejet (corp.eyejet.com)	Easy creation of video mail from a desktop or mobile device.
Vimeo (vimeo.com/)	A free video hosting service; Vimeo Plus is a fee-based premium version.
Dropbox (www.dropbox.com)	A smaller community than YouTube with a more serious focus and no ads.
Google Drive and Apps	Free file storage and syncing
Mover (mover.io/)	Google's free cloud storage and Web applications
Backupify (www.backupify.com)	Free cloud storage that can also move files between different cloud service
Amazon S3 (Simple Storage Service)	Backup of SaaS data from Google apps, email, and other cloud services
	Inexpensive mass storage from Amazon

Cloud Services

Many important services are in the cloud, and they are an important part of the Web 2.0 and mobile success story. Chapter 2 lists a number of cloud computing services, many of which do not fall into a neat category but are nonetheless very useful and worth exploring. There are books devoted to each of these tools as well as to the broader topic of how cloud computing and information behaviors are changing how we work. There are many useful tools for productivity applications, calendars, surveys and polling, and data storages. A few of these options are listed in Table 16.4; a source on other innovative technologies that support library 2.0 is by Courtney (2007).

CRITIQUES OF WEB 2.0 AND LIBRARY 2.0

Critics of Web 2.0 and library 2.0 are not necessarily antitechnology luddites and contrarians; there are thoughtful perspectives and cautionary views that should not be summarily dismissed. Some critiques are related to the oversimplification associated with the 2.0 label itself, whether it is applied to the Web or to libraries. Lankes (2011) expresses an understandable weariness with the label, especially when applied to libraries, noting that they have undergone many equally important transitions over a very long history. Casey and Savastinuk (2007) remind us that library 2.0 is not the only mission for libraries, quoting a manager who suggests that “though many library users have needs that have changed in step with technological innovations, many have not, and we do a disservice to those patrons if we focus exclusively on keeping up with technological innovation” (p. 6). Chant (2014) agrees, reminding us that the Pew Library Series studies all suggest that many patrons are “distinctly old-fashioned in their library use.”

Other critiques focus on the privacy issues that arise from 2.0 approaches, many of which are of particular concern to librarians whose commitment to patron privacy is enshrined in the ALA code of ethics. It is appropriate to review these concerns as well as to consider what Zimmer (2008) describes as other possible unintended consequences to Web 2.0 technologies:

Web 2.0 also embodies a set of unintended consequences, including the increased flow of personal information across networks, the diffusion of one's identity across fractured spaces, the emergence of powerful tools for peer surveillance, the exploitation of free labor for commercial gain, and the fear of increased corporatization of online social and collaborative spaces and outputs. (para. 2)

Certainly, privacy issues arise in many facets of Web 2.0. Every service you sign up for requests and stores personal information about you, and the service has unfettered access to the content its users produce or store. Privacy concerns seem to have a generational component; for that matter, many Web 2.0 services appear to have varying levels of appeal to different age groups. This could relate to a lack of technology skills to use the service, but it could also have more to do

with how different personalities and age groups view communication technology and personal space. Younger audiences appear to be much more comfortable with sharing vast amounts of personal information via social networks, while others find such public venues uncomfortable or invasive. In the past, some telephone customers wanting to protect their privacy would pay extra fees not to be listed in a directory; such an individual would seem an unlikely user of social media sites. In addition to privacy concerns, some users, probably more so in older demographic groups, might believe that social networking sites simply do not offer a service they desire or that is worth the investment of time. Such concerns could also apply to the use or nonuse of IM or Twitter; avoidance may have nothing to do with technological comfort or ignorance of its availability but might simply reflect the lack of a perceived need for that type of brief, immediate communication with others. After all, although their numbers are shrinking, there still are people who do not have or want cell phones and wonder who everyone else is constantly talking to as they walk or drive.

Another concern Web 2.0 brings to the fore is the displacement of the gatekeeper role of older media and its replacement by the “cult of the amateur.” Tenopir (2007) discusses Web 2.0 critic Andrew Keens’ book on this topic and finds some kernels of truth in his concern that reliable accurate information is being overwhelmed by unfiltered, unattributed, and unvetted sources or, in Keens’ words, “digital narcissism.” However, Tenopir notes that quality still matters, and she takes Keens’ message to be “buyer (reader) beware”—a sensible proposition that also reinforces a professional role in promoting information literacy skills.

Other critics observe that while pervasive technology can connect people as never before, people are more alienated than ever. Turkle (2007) develops this theme along several lines. She describes the tyranny of devices that require constant attention, taking time away from other forms of reflection or face-to-face interaction. People interrupt face-to-face conversations to answer a cell phone call or text message. Children grow up tethered to parents and peers by cell phones, and the emphasis on “bite size” messages and instant replies can preclude responses that are more thoughtful and reflective. Second Life allows a loner not to be alone but lacks the demands and rewards of genuine intimate friendships. Social networking entertains but can distract the young from real-world action requiring their actual engagement. Some critics suggest that Facebook is devaluing the nature of friendship itself, Deresiewcz (2009a) refers to it as “faux friendship.” He also laments the “end of solitude” and the culture of celebrity being brought about by constant and ubiquitous connectivity (Deresiewcz, 2009b). Even Web 2.0 guru Tim O'Reilly (2006) has expressed concerns about the dark side of these technologies; he listed three of his concerns in a speech at the University of California School of Information:

First, privacy. Collective intelligence requires the storage of enormous amounts of data. And while this data can be used to deliver innovative applications, it can also be used to invade our privacy . . .

Second, concentration of power. While it's easy to see the user empowerment and democratization implicit in web 2.0, it's also easy to overlook the enormous power that is being accrued by those who've successfully become the repository for our collective intelligence. Who owns that data?

Is it ours, or does it belong to the vendor? If history is any guide, the democratization promised by Web 2.0 will eventually be succeeded by new monopolies, just as the democratization promised by the personal computer led to an industry dominated by only a few companies . . . Don't just take for granted that technology will bring us a better world. We must engage strenuously with the future, thinking through the dark side of each opportunity, and working to maximize the good that we create while minimizing the harm.

Third, greed. Web 2.0 has ignited a new feeding frenzy among venture capitalists and entrepreneurs. It's perhaps too early to call it a bubble, but once again, enormous fortunes are being created by people with little more than a bright idea and an instinct for how to harness the power of new technology. (para. 39–41)

Concerns about the negative impact of technology on culture and people are not new; in hindsight, some appear to have been exaggerated and others may just reflect generational shifts. Douglas Adams (2002) suggests three rules on our reactions to technology: anything in the world when you're born is normal; anything developed between ages 15 and 35 is new; and anything invented after you're 35 is against the natural order of things (p. 95). Even so, writers such as Carr (2008b) still raise questions worth pondering about how the technologies and media we use can shape how we read and think. However, counterarguments to that view certainly can be made that highlight the positive outcomes possible from the knowledge sharing, connectivity, and communication that these technologies promote. The global connections enabled by Web 2.0 technologies could translate into broader awareness and commitment to solving real-world problems. Children may be reading fewer traditional materials, but they are reading and writing a great deal online. Not all the seemingly 24/7 connections between parents and children result in intrusive helicopter parents but instead can result in closer bonds and more open communication between them. Nonetheless, like all technologies that have come before, there are tradeoffs associated with their use, and both sides of the conversation have merits.

These tradeoffs are apparent in many of the technological choices facing libraries. The Internet of everything could eventually include every item in a library collection; sensors in books could go beyond current RFID technologies and constantly track their location 24/7, eliminating the need for checkouts, but at the expense of patron privacy. The use of cloud and grid computing offer both efficiencies and potential cost savings, but with loss of local control and added concerns about data security. Libraries could use collaborative tools to engage their communities by serving both as a publisher of their stories and as a deep archives of those community memoirs and ruminations, but with a cost of time and resources. eBooks represent an increasingly important resource, but with the cost of supporting multiple devices and formats, as well as dealing with publisher licensing agreements for these materials. Emerging Web technologies, mobile solutions, microsensors, and social media offer both opportunities and challenges. The broader debates about technologies' effects on society as well as concerns about privacy and data security are important and will certainly continue, but they should not deter libraries from exploring

and embracing new technologies as they seek to fulfill their mission and serve their communities.

SUMMARY

The statement by Farkas (2007) that “now is an amazing time to be a librarian” is still true today (p. 43). The Internet and Web in all its forms enable many exciting possibilities, while simultaneously presenting significant challenges. There are new tools that can be used to enhance library online services and engage users in new forms of participation and content development. Core technologies such as the ILS and library catalogs are evolving to better serve digital collections and user needs. Social media can build new, dynamic communities that can be leveraged to benefit the institution and its community. The mobile Web is changing information and entertainment behaviors, requiring a proactive response by libraries. The Web and digital collections have dramatically changed the physical design and utilization of library spaces. The Maker culture is influencing the creation of new learning spaces and decisions about purchasing new technologies such as 3D printers that could support them. A major challenge is the difficulty in keeping up with new trends and technologies, while remaining aware of O'Reilly's advice for “thinking through the dark side of each opportunity” (2006, para. 43). Privacy concerns, cloud data security, and resource allocation for emerging technologies represent potential downsides to be considered. Librarians must continue to strive to embrace the new without abandoning traditional functions and audiences, striking a delicate balance between being an early adopter/trend-spotter and a skeptic, exploring new technologies for their own sake but ultimately focusing on the value-added benefits of their use.

REFERENCES

- Abram, S. (2005). The Google opportunity. *Library Journal*, 130(2).
- Abram, Stephen. (2009, September 9). APIs and libraries. Retrieved August 30, 2013, from <http://stephenslighthouse.com/2009/09/01/apis-and-libraries/>
- Adams, Douglas. (2002). *The salmon of doubt: hitchhiking the galaxy one last time* (1st ed.). New York: Harmony Books.
- Allen, M. (2008). Web 2.0: An argument against convergence. *First Monday*, 13(3).
- Ally, Mohamed, & Needham, Gill. (2012). *M-libraries: Transforming libraries with mobile technology*. London: Facet Publishing.
- Alpcan, T., Bauckhage, C., & Kotsovinos, E. (2007). Towards 3D Internet: Why, what, and how? Retrieved August 28, 2013, from http://www.kotsovinos.com/research/papers/Alpcan-3D_Internet.pdf.
- Anderson, Chris, & Wolff, Michael (2010, September). The Web is dead. Long live the Internet. *Wired*, 11.
- Bates, M. J. (2002). *Toward an integrated model of information seeking and searching (Keynote)*. Paper presented at the Fourth International Conference on Information Needs, Seeking and Use in Different Contexts, Lisbon, Portugal.
- Bell, L., Pope, K., Peters, T., & Galik, B. (2007). Who's on third in Second Life? *Online*, 31(4).

- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.
- Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic Web. *Scientific American*, 284(5), 10.
- Berlot, J. C., McClure, C. R., Wright, C. B., Jensen, E., & Thomas, S. (2009). Public libraries and the Internet 2009: Study results and findings. Retrieved May 14, 2014, from http://www.ii.fsu.edu/content/download/16874/109694/03_executive%20summary1-7.pdf.
- Bosak, J., & Bray, T. (1999). XML and the second-generation Web. *Scientific American*, 280(5), 89–94.
- Breeding, M. (2007). Next-generation library catalogs. *Library Technology Reports*, 43(4), 5–42.
- Breeding, M. (2014). Library Systems Report 2014. *American Libraries*, 45(5), 21–33.
- Brenner, Joanna, & Smith, Aaron. (2013, August 5). 72% of Online adults are social networking site users. Retrieved September 5, 2013, from <http://pewinternet.org/Reports/2013/social-networking-sites.aspx>.
- Bugeja, M. J. (2007). Second thoughts about Second Life. *Chronicle of Higher Education*, 54(3), C2–C4.
- Calhoun, K. (2006, March 17). The changing nature of the catalog and its integration with other discovery tools. Retrieved November 18, 2013, from <http://www.loc.gov/catdir/calhoun-report-final.pdf>.
- Carr, N. (2008a). *The big switch*. New York: W.W. Norton & Company.
- Carr, N. (2008b, July/August). Is Google making us stupid? *Atlantic Monthly*, 302(1), 56.
- Casey, M. E., & Savastinuk, L. C. (2007). *Library 2.0: A guide to participatory library service*. Medford, NJ: Information Today.
- Chant, I. (2014). Research: Pew Concludes Library Series. *Library Journal*, 139(8), 19–19.
- Christensen, C. M. (2003). *The innovator's dilemma: The revolutionary book that will change the way you do business*. New York: Collins.
- Ciocco, R., & Huff, A. (2007). Mission IM-possible: Starting an instant message reference service using Trillian. *Computers in Libraries*, 27(1), 26–31.
- Clark, Jason A. (2012). *Building mobile library applications*. Chicago, IL: Library and Information Technology Association.
- Coleman, J., Katz, E., & Menzel, H. (1957). The diffusion of an innovation among physicians. *Sociometry*, 20(4), 253–270.
- Coombs, Karen A., & Hollister, Amanda J. (2010). *Open source Web applications for libraries*. Medford, NJ: Information Today, Inc.
- Cote, Denise, Kraemer, Beth, Nahl, Diane, & Ashford, Robin. (2012). Academic librarians in Second Life. Retrieved August 26, 2013, from http://uknowledge.uky.edu/libraries_facpub/17.
- Courtney, Nancy. (2007). *Library 2.0 and beyond: Innovative technologies for tomorrow's user*. Westport, CT: Libraries Unlimited.
- Coventry, M. (2006). Libraries for a new generation: Getting the net gen on the right information highway. *UMNNews*. Retrieved October 1, 2007, from http://www1.umn.edu/umnnews/Feature_Stories/Libraries_for_a_new_generation.html.
- Coyle, K. (2007). The library catalog in a 2.0 world. *The Journal of Academic Librarianship*, 33(2), 289–291.
- Dankowski, Terra. (2013). How libraries are using social media. *American Libraries*, 44(5), 38–41.
- Deresiewicz, William. (2009a, December 6). Faux friendship. Retrieved September 29, 2010, from <http://chronicle.com/article/Faux-Friendship/49308/>

- Deresiewicz, William. (2009b, January 30). The end of solitude. Retrieved September 5, 2013, from <http://chronicle.com/article/The-End-of-Solitude/3708>.
- Desilver, Drew. (2013, May 20). 5 Facts about Tumblr. Retrieved September 4, 2013, from <http://www.pewresearch.org/fact-tank/2013/05/20/5-facts-about-tumblr/>
- Dibbell, J. (February, 2008). Griefer madness. *Wired*, 16, 90–97.
- EDUCAUSE. (2008). 7 Things you should know about Flickr. Retrieved May 30, 2008, from <http://net.educause.edu/ir/library/pdf/ELI7034.pdf>.
- Engard, Nicole C. (2009). *Library mashups: Exploring new ways to deliver library data*. Medford, NJ: Information Today, Inc.
- Enis, M. (2014). Putting the Pieces Together. *Library Journal*, 139(6), 32–32.
- Farkas, M. (2007). Balancing the online life. *American Libraries*, 38(1), 42–45.
- Fay, Robin M., & Sauers, Michael. (2012). *Semantic Web technologies and social searching for librarians*. Chicago, IL: ALA TechSource.
- Fichter, D. (2006). Using wikis to support online collaboration in libraries. *Information Outlook*, 10(1), 30–33.
- Gartner says 80 percent of active Internet users will have a “Second Life” in the virtual world by the end of 2011. (2007, April 24). Retrieved August 28, 2013, from <http://www.gartner.com/newsroom/id/503861>.
- Gartner's 2009 hype cycle special report evaluates maturity of 1,650 technologies. (2009, August 11). Retrieved August 28, 2013, from <http://www.gartner.com/newsroom/id/1124212>.
- Georgas, H. (2013). Google vs. the library: Student preferences and perceptions when doing research using Google and a federated search tool. *Libraries and the Academy*, 13(2), 165–185. Retrieved January 7, 2014, from Project MUSE database.
- Gill, J. (2008, January 17). Researchers' web use could make libraries redundant. Retrieved May 1, 2008, from <http://www.timeshighereducation.co.uk/story.asp?storycode=400168>.
- Google. (2008). Mission statement. Retrieved May 1, 2008, from <http://www.google.com/corporate>.
- Griffey, Jason. (2010). *Mobile technology and libraries*. New York: Neal-Schuman Publishers.
- Hafner, K. (2007, August 19). Seeing corporate fingerprints in Wikipedia edits. *New York Times*, p. 1.
- Hampton, Keith N., Lauren Sessions Goulet, Lee Rainie, Kristen Purcell. (2011, June 16). Social networking sites and our lives. Retrieved May 20, 2013, from <http://www.pewinternet.org/Reports/2011/~/media/Files/Reports/2011/PIP%20-%20Social%20networking%20sites%20and%20our%20lives.pdf>.
- Hawking, D., & Zobel, J. (2007). Does topic metadata help with Web search? *JASIST*, 58(5), 613–626.
- Hempel, Jessi, Mansour, Iris, & Southward, Brandon. (2013, September 16). Social media all-stars. *Fortune*, 168, 133–139.
- Jones, S., & Madden, M. (2002, September 15). The Internet goes to college. *Internet and American Life Project*. Retrieved October, 2007, from http://www.pewinternet.org/pdfs/Pip_College_Report.pdf.
- Jowitt, A. L. (2008). Creating communities with podcasting. *Computers in Libraries*, 28(4), 14–15, 54–16.
- Kaplan, Andreas M., & Haenlein, Michael. (2010). Users of the world, unite! The challenges and opportunities of social media. *Business Horizons*, 53(1), 59–68. doi: <http://dx.doi.org/10.1016/j.bushor.2009.09.003>.
- Kelley, K., Leatherman, C.C., & Rinna, G. (2013). Is it really time to replace your ILS with a next-generation option? *Computers in Libraries*, 33(8), 11–15.

- Kestenbaum, D. (2008). Old drug offers new hope for Marfan syndrome. Retrieved May 30, 2008, from <http://www.npr.org/templates/story/story.php?storyId=90257827>.
- KZERO Worldwide. (2012, January 6). Virtual worlds: Industry and user data. Retrieved September 29, 20113, from http://www.slideshare.net/nicmitham/kzero-universe-q4-2011?from=ss_embed.
- La Counte, Scott. (2012). *Going mobile: Developing apps for your library using basic HTML programming*. Chicago, IL: ALA Editions.
- Lankes, R. David. (2011). *The atlas of new librarianship*. Cambridge, MA: MIT Press.
- Madden, Mary. (2010, August 27). Older adults and social media. Retrieved September 3, 2013, from <http://www.pewinternet.org/Reports/2010/Older-Adults-and-Social-Media.aspx>.
- Madden, Mary, Lenhart, Amanda, Cortesi, Sandra, Gasser, Urs, Duggan, Maeve, Smith, Aaron, & Beaton, Meredith. (2013, May 21). Teens, social media, and privacy, from <http://www.pewinternet.org/Reports/2013/Teens-Social-Media-And-Privacy/Main-Report/Part-1.aspx>.
- Mann, T. (2006, April 3). The changing nature of the catalog and its integration with other discovery tools: A critical review. Retrieved November 18, 2013, from <http://www.guild2910.org/AFSCMCECalhounReviewREV.pdf>.
- Mi, J., & Weng, C. (2008). Revitalizing the library OPAC: Interface, searching and display challenges. *Information Technology and Libraries*, 27(1), 5–22.
- Michel, Jason Paul. (2012). *Web service APIs and libraries*. Chicago IL: Amercian Library Association.
- Mitchell, J. (2014). Beyond the Maker Space. *Library Journal*, 139(9), 37–37.
- Murthy, Dhiraj. (2013). *Twitter: Social communication in the Twitter age*. Cambridge: Polity.
- Nielsen Company. (2008, May 14). Wikipedia U.S. Web traffic grows 8,000 percent in five years, driven by search. Retrieved June 10, 2008, from http://www.nielsen-netratings.com/pr/pr_080514.pdf.
- Online Community Library Center (OCLC). (2006, June 1). OCLC reports on college students' library perceptions. Retrieved May 1, 2008, from <http://www.libraryjournal.com/info/CA6340281.html>.
- O'Reilly, T. (2005, October 1). Web 2.0: Compact definition? Retrieved October 1, 2007, from <http://radar.oreilly.com/archives/2005/10/web-20-compact-definition.html>.
- O'Reilly, T. (2006, May 14). My commencement speech at SIMS. Retrieved June 2, 2008 from <http://radar.oreilly.com/archives/2006/05/my-commencement-speech-at-sims.html>.
- Peters, T., & Bell, L. (2008, April 11). Trends, fads or folly: Spotting the library trends that really matter. Retrieved April 15, 2008, from <http://www.collegeofdukepress.com/library-learning-network/soaring-to-excellence-2008/spotting-the-library-trends-that-really-matter/>
- Pressman, Aaron (2014, January 16). More than 3 million teens flee Facebook. Retrieved January 19, 2014 from <http://finance.yahoo.com/blogs/the-exchange/about-11-million-fewer-likes-for-facebook-among-the-teenage-set-154852067.html>.
- Purcell, Kristen, Brenner, Joanna, & Rainie, Lee (2012, March 9). Search engine use 2012. Retrieved September 11, 2013, from <http://www.pewinternet.org/Reports/2012/Search-Engine-Use-2012.aspx>.
- Purvis, M., Sambells, J., & Turner, C. (2006). Beginning Google maps applications with PHP and AJAX; from novice to professional. Berkeley, CA: Apress.
- Rainie, L. (2005, November 20). Search engine use shoots up in the past year and edges towards email as the primary internet application. *Internet and*

- American Life*. Retrieved May 1, 2008, from http://www.pewinternet.org/PPF/r/167/report_display.asp.
- Rainie, L. (2007, January 31). Tagging play: Forget Dewey and his decimals, Internet users are revolutionizing the way we classify information—and make sense of it. *Pew Internet & American Life Project*. Retrieved May 30, 2008 from <http://pewresearch.org/pubs/402/tagging-play>.
- Rapacki, S. (2007). Why teens need places like MySpace. *Young Adult Library Services*, 5(2).
- Reiss, S. (2008, May). Planet Amazon. *Wired*, 16, 88–95.
- Rethlefsen, M. L. (2007). Tags help make libraries DELICIO.US. *Library Journal*, 132(15), 26–28.
- Saul, D. (2014, January 15). 3 Million teens leave Facebook in 3 years: The 2014 Facebook demographic report. Retrieved January 19, 2014, from <http://istrategylabs.com/2014/01/3-million-teens-leave-facebook-in-3-years-the-2014-facebook-demographic-report/>
- Schull, Diantha Dow. (2013). *50+ Library services: Innovation in action*. Chicago, IL: ALA Editions.
- SFX—The OpenURL link resolver and much more. (2012). Retrieved September 16, 2013, from <http://www.exlibrisgroup.com/category/SFXOverview>.
- Shannon, V. (2006, May 24). A “more revolutionary” Web. Retrieved May 1, 2008, from <http://www.iht.com/articles/2006/05/23/business/web.php>.
- SMS triples in three years. (2011). Retrieved September 19, 2013, from <http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf>.
- Soloman, Laura. (2013). *The librarian’s nitty-gritty guide to social media*. Chicago, IL: ALA Editions.
- Stephens, M. (2007). *Web 2.0 & libraries, part 2: Trends and technologies* (Report). Chicago, IL: ALA.
- Sullivan, D. (2005, January 28). Search engine sizes. Retrieved October 1, 2007, from <http://searchenginewatch.com/showPage.html?page=2156481>.
- Tam, Donna. (2013, January 30). Facebook by the numbers: 1.06 billion monthly active users. Retrieved May 20, 2013, from http://news.cnet.com/8301-1023_3-57566550-93/facebook-by-the-numbers-1.06-billion-monthly-active-users/
- Tennant, Roy. (2011, March 17). While web 2.0 (and the related “library 2.0” idea) still receives lots of attention, it has lost some of its “cutting edge” mystique. Retrieved September 27, 2012, from http://blog.libraryjournal.com/tenantdig_itallibraries/2011/03/17/7-words-or-phrases-to-never-say-or-write-again/
- Tenopir, C. (2007). Web 2.0: Our cultural downfall? *Library Journal*, 132(20), 36.
- Top 15 most popular social networking sites. (2013, August). Retrieved August 14, 2013, from <http://www.ebizmba.com/articles/social-networking-websites>.
- Torrone, P. (2011, March 10). Is it time to rebuild & retool public libraries and make “TechShops”? Retrieved May 16, 2014, from <http://makezine.com/2011/03/10/is-it-time-to-rebuild-retool-public-libraries-and-make-techshops/>
- Turkle, S. (2007). Can you hear me now? *Forbes.com*. Retrieved June 4, 2008, from http://www.forbes.com/forbes/2007/0507/176_print.html.
- Turnau, Amber. (2012, July 13). The 5 C’s of social media. Retrieved September 17, 2013, from <http://www.6smarketing.com/blog/5-cs-of-social-media-marketing/>
- U.S. Census. (2006). Computer and Internet use in the United States: 2003. Retrieved October 1, 2007, from <http://www.census.gov/population/pop-profile/dynamic/Computers.pdf>.
- W3C. (2001). Semantic Web activity. Retrieved October 1, 2007, from <http://www.w3.org/2001/sw>.

- Webber, D., & Peters, A. (2010). *Integrated library systems: Planning, selecting, and implementing*. Santa Barbara, CA: Libraries Unlimited.
- Weber, J. (2006). Evergreen: Your homegrown ILS. *Library Journal*, 131(20), 38–41.
- Zeitgeist overview. (2013). Retrieved September 6, 2013, from <http://www.librarything.com/zeitgeist>.
- Zimmer, M. (2008). Preface: Critical perspectives on Web 2.0. *First Monday*, 13(3).
- Zitrain, Jonathan. (2008). *The future of the Internet and how to stop it*. New Haven, CT: Yale University Press.

ADDITIONAL READING AND LINKS

- Information behaviour of the researcher of the future. Retrieved February 1, 2008, from <http://www.bl.uk/news/pdf/googlegen.pdf>.
- Miller, W., & Pellen, R.M. (Eds.). (2005). *Libraries and Google*. Binghamton, NY: Haworth Information Press.
- O'Reilly, T. (2005, September 30). What is Web 2.0: Design patterns and business models for the next generation of software. Retrieved May 1, 2008, from <http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>.
- Scholz, T. (2008). Market ideology and the myths of Web 2.0. *First Monday*, 13(3).
- Silver, D. (2008). History hype and hope: An afterword. *First Monday*, 13(3).

About Makerspaces: (<http://makerspace.com/>)

A Librarian's guide to Makerspaces (<http://oedb.org/ilibrarian/a-librarians-guide-to-makerspaces/>)

This page intentionally left blank

Appendix 1

The Binary Machine

BINARY NUMBERS

To understand why the computer is a binary machine, it is necessary to review binary numbers and Boolean logic. A firm understanding of these basic concepts is essential to a number of topics throughout this text, such as IP addressing, subnet masks, the ASCII byte streams used in all Internet protocols, and the encoding schemes used to handle binary file formats.

Number Systems: Base 10, Base 2, and Base 16

People are most familiar with the decimal (base 10) numbering system and instinctively understand their values. Our comfort level with this system is due to a variety of cultural, political, and biological factors (for instance, people have 10 digits to help with counting and U.S. currency uses decimal, thanks to Alexander Hamilton). A number system uses symbols (digits) to represent values and assigns different meanings to the digits depending on their *position* in the number. It is possible to have numbering systems that ignore “place.” Simple counting systems such as groups of five slashes (four vertical with a fifth drawn through the others) or even more complex Roman numerals, do not use column place in this way, but Western numbering does. Base-10 numbers require 10-digit symbols and a shared understanding that the position of the digit reflects some power of 10 as the column value. The 10 digits are 0–9 (remember that zero is a perfectly good number!) When you view a base-10 number such as 7,826, it is immediately parsed without much thinking as seven thousands, eight hundreds, two tens, and six ones. That parsing presumes, however, this is a base-10 number; to be more complete when writing

426 Appendix 1

this number, technically information about its base with the subscript (10), as in $7,826_{(10)}$ should be provided. Many numbering systems are possible in addition to base-10, and they are all equal in the sense that any value can be represented in them. The difference is only in the column values and the number of digits used.

The 10 symbols required for base-10 numbers present an engineering problem to the designers of computing systems. A device that discriminates 10 different levels of magnetic charges and associates each of them with 10 different digits is hard to build. It is, however, relatively easy to create devices that can discern two different states, such as a switch that is on or off, or the presence or absence of a charge, and associate each state with the digits 0 or 1. This is why the binary (base-2) numbering system is a great fit for computers. Only two digits are required (0 and 1), and the column position values are based on powers of 2. Table A1.1 shows a base-10 example that analyzes the number $7,826_{(10)}$.

To determine the value this number represents, note that the position of the digit adds a factor of 10 to the column value. Binary numbers work the same way. For example, try to convert the binary number $100101_{(2)}$ (the subscript tells you not to view this as a base-10 number) to decimal. Table A1.2 shows this number's value.

The binary number 100101 is equivalent to decimal 37. Using the same approach you could reverse the process to determine that the decimal number 7,826 is equal to the binary number 1110 001110110. It is apparent from this exercise that although the total values are the same, it takes a longer number to express a given value in binary compared to base-10; this makes sense, as you see that each column “packs more punch” with base-10. Note, too, that binary is hard for people to read, and to make the long 12-digit base-2 number more readable, a 4-bit portion is separated from the 8-bit portion. Putting a group of 8 bits together is significant as this octet represents a *byte*, which in

TABLE A1.1 A Base-10 Number. Column position values are based on powers of 10; starting at the far right, the exponent is zero. The exponent value is incremented by one with each column move to the left. Note that any number raised to the zero is one, and any number raised to the one is the number itself.

Power of 10	10^3	10^2	10^1	10^0
Column value	7×1000	8×100	2×10	6×1
Number	7	8	2	6

TABLE A1.2 Binary Number 100101. The column values are all powers of 2 where the exponent value starts with 0 at the far right and is incremented by 1 with each column to the left.

Power of 2	2^5	2^4	2^3	2^2	2^1	2^0
Column value	1×32	0×16	0×8	1×4	0×2	1×1
Number	1	0	0	1	0	1

turn is usually associated with a single character in various data representation schemes such as ASCII (the 4-bit portion is half a byte or a *nibble*).

Another numbering system that is important in computing is the hexadecimal numbering system (abbreviated as hex), which is base 16. Base 16 requires 16 digits; 0–9 gives 10, but then there are no more numbers. However, because these are just symbols that are associated with a value, any symbol can stand in for a number. Hex uses the letters a-f for the needed six additional digits (for 10, 11, 12, 13, 14, and 15). Because base 16 numbers use 16 as the base, the value of each column moving to the left increases very quickly, as shown in Table A1.3.

This rapid increase in column values with hex makes representing very large numbers very compact. Also, from the comparison with base 2, note that the hex column values map directly to base 2 column values if each base 2 column exponent value is incremented by 4 ($2^4 = 16$); this makes the conversion between binary and hex slightly easier than the conversions to decimal.

Data Representation Codes

From the discussion so far, the following conclusions can be made:

- There are many ways to represent a value.
- Computers only use binary numbers, but people prefer not to work with these very long numbers; they are usually converted to base 10 or base 16 to simplify their notation for our use.
- The same value expressed in binary requires a longer number (i.e., more digits) than when the same value is expressed in either base 10 or base 16.

The problem of how to represent character text in computers was addressed early in the development of digital computer systems. In the nineteenth century, Samuel Morse addressed a similar problem for the telegraph technology of that era. Morse code used dots and dashes (essentially a binary code) to represent all the characters of the Western alphabet. The ASCII code serves as a standard for interpreting computer text formats. The designers of the code needed to define how many bits are required to identify all the possible characters in the Western alphabet. There are 26 letters, and each has upper and lowercase forms. There is also a need for punctuation characters and perhaps special symbols (such as \$, %, @). In a code with just two states (0 or 1), the number of unique values is dependent on how many digits long the code

TABLE A1.3 Hex Numbering Compared to Base 2

Power of 16	16^5	16^4	16^3	16^2	16^1	16^0
Valuee	1,048,576	65536	4096	256	16	1
Power of 2	2^{20}	2^{16}	2^{12}	2^8	2^4	2^0

can be. A code that is just three digits long can have eight (or 2^3) possible values: 000, 001, 010, 100, 011, 110, 101, and 111. A heuristic can be deduced that the number of possible values (or addresses) is 2 raised to a power that is equal to the number of bits long the code is. Standard ASCII uses 7 bits, making it possible to represent 128 (or 2^7) different symbols. Extended ASCII uses all 8 bits of a byte, allowing for 256 symbols. This is more than adequate for the Western alphabet, but not for many other character sets. Unicode comes in 16- and 32-bit versions; with 16 bits, 2^{16} or 65,536 code values are possible.

Binary Addressing Schemes

Computer and network systems use binary numbers to address elements such as RAM addresses, MAC codes for network interface cards, and IP addresses for Internet hosts. The same simple rules apply; binary numbers are used for all such addresses, and the number of possible addresses depends on the length of the permitted address number in bits. IPv4 uses 32-bit addressing, which means the theoretical limit for these addresses would be at most 2^{32} , or just over 4 billion (it turns out that there are additional constraints on these addresses as discussed in Chapter 4). MAC addresses use a 48-bit number, usually displayed in a hex format for a more compact representation.

BOOLEAN LOGIC

The other reason computers are binary machines is that they depend on a binary logic system called Boolean logic, developed by George Boole in 1847, thereby enabling librarians to start using Boolean logic in their searches! Of course, those IR applications actually came much later, but Boolean searching is a convenient example to demonstrate this algebraic system.

A search using two terms, for instance “cat” and “dog,” could be combined with Boolean operators. A Boolean AND means a retrieval takes place only when both terms are in a record, with a Boolean OR, either term (or both) is sufficient to result in a retrieval. This is represented algebraically as a logic truth table for the Boolean operators AND, OR, and NOT. Instead of words, truth tables use numbers to represent the operands. The Boolean AND results are shown along with the Boolean OR in Table A1.4. Table A1.5 shows the Boolean NOT results.

TABLE A1.4 The Boolean AND and Boolean OR Truth Tables

AND Table

Operand	Operand	Result
0	0	0
0	1	0
1	0	0
1	1	1

OR Table

Operand	Operand	Result
0	0	0
0	1	1
1	0	1
1	1	1

TABLE A1.5 The Boolean
NOT Operation

Operand	Result
0	1
1	0

The Boolean AND looks like the arithmetic operation of multiplication, the OR looks like the arithmetic operation of addition, and Boolean NOT is an inverse function. This simple algebraic system is the foundation of the logic gates built into the circuitry of a computer. This logic system is explained in other areas of this text, such as in the discussion of subnet masks.

This page intentionally left blank

Appendix 2

Web Hosting

The platform that houses the website creates a number of potential issues for the Web author, and an awareness of the idiosyncrasies of the operating system in use and the availability of other server-side programs is needed in the design and implementation of a site. For example, case sensitivity may or may not be an issue on a particular host. In addition, a website designed to utilize Microsoft's ASP technology requires the installation of the Microsoft SSI components, so it is important to know if these are available prior to committing to that approach.

UNIX WEB SERVERS

Many large-scale Web servers are UNIX or Linux (a PC version of UNIX) systems; this is the case for a number of important reasons:

1. Historically, UNIX was the preferred OS for the high-end workstations that were available when the Web was developed.
2. UNIX was an early multiuser, multitasking operating system that could handle the large number of input/output operations such servers must perform.
3. UNIX “scales up” well to accommodate the large number of simultaneous users often accessing such a server compared to alternative operating systems.
4. UNIX can provide a reasonably secure operating environment.
5. UNIX has a large, well-defined support and developer community.

Many Web designers and authors therefore find they must work with a UNIX host as a Web server at some point in their careers, so a basic knowledge of some UNIX features as they affect Web authoring is useful. This UNIX discussion is limited to that perspective; clearly, the administrator would have many responsibilities that require greater expertise. Those needing coverage that is more detailed should consult one of the many excellent UNIX sources (Kaplenk, 1999; Martin, Prata, Waite, Wessler, & Wilson, 2000).

First, unlike Windows systems, UNIX is a case-sensitive environment; consequently, directory and file name references in HTML code must be written with this in mind. Links and image references that work fine on a local Windows PC will fail when uploaded to the UNIX host if there are case mismatches between source code and actual names. The adoption of a consistent directory and file-naming convention such as always using lowercase for all filenames and extensions can help alleviate this problem.

Another feature of UNIX is directory and file-level security. Understanding permissions and their application is necessary for the successful delivery of Web content from a UNIX host. Failure to assign the correct permission array to both the file and the directory (i.e., folder) that houses it results in the HTTP error message “FORBIDDEN—you don’t have permission to view this file,” or, if the permissions are incorrect for an image, the “broken image” icon appears in the browser view. Although these errors can result from other causes, directory and file permissions that are incorrect are mostly the likely reasons.

UNIX Permissions

There are three permissions that can be set on or off for a directory or file; they are **read** (you can view the file), **write** (you can write to the file), and **execute** (if it is executable, you can execute it). Each of these permissions can be on or off for different users, depending on who needs access and why. If there are two states (on or off) and three permissions (r, w, and x), the number of combinations is then 2^3 or 8, ranging from all three permissions denied to all three granted. The eight combinations of possible permissions are referred to as the **octal value** for a specific permission assignment. If each of the eight combinations of permissions is assigned a number starting with 0 and ending with 7, the octal values that can be associated with each permission setting are as follows: 0 = none, 1 = execute, 2 = write, 3 = write + execute, 4 = read, 5 = read + execute, 6 = read + write, 7 = read + write + execute. These combinations are summarized in Figure A2.1.

UNIX also identifies three entities when it comes to whom permissions are granted. UNIX systems have many users, and the first entity is the **user** who has rights to the directories and files they own. Second, the entity **groups** are a reflection of the fact that multiuser systems

Mode	Octal Value
---	0
--X	1
-W-	2
-WX	3
r--	4
r-X	5
rW-	6
rWX	7

Figure A2.1 UNIX permission modes and their respective octal values.

typically allow the creation of workgroups who share certain privileges. UNIX administrators routinely create groups of users, so certain types of access can be granted to the group as a whole. Finally, there is the entity **others**, referring to the rest of the world who might access this system. Why should you care about these others, whoever they are? It is because when you wish to publish Web content, UNIX needs explicit permission to share those files to everyone's browser. The three entities—user, groups, and others—are abbreviated in commands with the first letter of each word: **u**, **g**, and **o**.

To set file permissions, the system requires information about *what* permissions are granted and to *whom* they are given. The general rule of thumb is to grant both **read** and **execute** permissions to both **groups** and **others** while retaining **read**, **write**, and **execute** for the **user**. Execute permission is needed for directories but not always for files unless the page contains scripting—in that case execute permission is required. Write access is restricted to the user or perhaps to a specific group but never granted to the entity “others.” If the file (or directory) permissions are not correct, the UNIX server does not send the requested file and instead responds with the “Forbidden” message as shown in Figure A2.2.

There are several ways to modify file permissions when needed. Most FTP clients have this feature, often from a menu initiated by right clicking on the file or directory. The UNIX host can also be accessed via a telnet client and the **chmod** (for “change mode”) command issued. For example, if a file named “**main.html**” needs to have the permissions set, the following command syntax should be used:

```
chmod 755 main.html
```

The 755 octal values grant the user read, write, and execute (7), groups read and execute (5), and others read and execute (5). Hence, the “755” octal setting is the “mantra” of Web file permissions; it is the common default for Web directories and files.



Forbidden

You don't have permission to access
/~jbmill00/jmiller.htm on this server.

Apache/2.2.15 (Red Hat) Server at sweb.uky.edu Port 80

Figure A2.2 Server “Forbidden” response when a requested file does not have read permission granted to others.

REFERENCES

- Kaplenk, J. (1999). *UNIX system administrator's interactive workbook*. Upper Saddle River, NJ: Prentice-Hall.
- Martin, D., Prata, S., Waite, M., Wessler, M., & Wilson, D. (2000). *UNIX primer plus* (3rd ed.). Indianapolis: Sams.

Glossary

absolute reference: Reference that unambiguously defines a location, such as a complete URL of an Internet resource.

addressing granularity: The level at which index terms are tracked in a document; terms can be associated with just a document ID, a block within a document, or the exact word position within the document.

algorithm: A series of steps by which a problem can be solved; usually used in the context of programming directives.

aliasing: A distortion or artifact of the sampling process when converting analog data to digital.

amplitude: The measure of a waveform height.

amplitude modulation: When information is added to a carrier signal through signal modulation.

anti-aliasing: A technique to compensate for aliasing by eliminating distortion. Graphics programs use this feature to smooth out the edges of images and text.

API (Application Program Interfaces): Programming code available to developers that allow for many site enhancements with just a small amount of programming skill. APIs are used to create various mashups that can perform many functions and provide enhancements to websites. APIs for apps can connect them to device functions such as the accelerometer, camera, and storage or be used for geolocation and media playback.

app: A program developed primarily for mobile devices that is usually designed to perform a specific task.

applet: Java programs called within a Web page, retrieved, and then run within the browser by the Java VM.

Archie index: Short for “archives,” it was developed at McGill University and was the first widely used Internet index; it provided access to files stored on anonymous FTP sites.

ARP (Address Resolution Protocol): A broadcast network message that seeks a MAC address associated with an IP address.

ARPANET (Advanced Research Projects Agency Network): A network initiative supported by DARPA. It was the first packet-switching data communications network and gave rise to the modern Internet.

array: A type of variable that stores multiple values in a tabular form.

ASCII (American Standard Code for Information Interchange): A text code that uses 7 or 8 bits to represent textual data. Standard 8-bit ASCII can represent 256 different symbols.

ASP (Active Server Pages): Web pages with embedded scripts that are processed on a server using the SSI programs. The ASP technology has been coupled with the Microsoft.NET initiative, a broad framework utilizing XML, SOAP, and other technologies to create an Internet-based platform of Web services for Windows systems

associative array: An array is a way to store multivalue variables using a two-column table. The left column is a series of string names that comprises the array index.

asymmetric: In the context of connection technologies, the difference between upstream and downstream bandwidth that makes a connection asymmetric.

authorities: In the HITS model, authorities are pages with a high number of in-degree links.

bandwidth: The capacity of the medium to carry a signal.

Big-S Semantic Web: The original W3C vision for a semantically enhanced Web that is based on XML, RDF, and OWL W3C standards. The approaches to creating a semantically enhanced Web have broadened to include other ways of creating and exploiting a “web of data.”

binary: Base 2 numbering that uses only zeros and ones.

binary independence retrieval (BIR): Retrieval model that uses a probability calculation to algorithmically estimate relevance.

bitmap images: See raster images.

block-level elements: In HTML, elements that are always rendered on a new line in the browser display.

blogs: From the words Web and log or a “weblog.” A personal narrative intended to be syndicated using RSS.

Boolean logic: An algebraic system developed by George Boole in 1847 that uses binary operands and three operators: AND, OR, and NOT. It is the foundation of computer logic gates and is also used as an IR model.

BOOTP: A protocol used to assign an IP address with a MAC address. In BOOTP, the computer booting up on the network broadcasts a request message with its MAC address, and the BOOTP server responds with the IP address as well as other network configuration information that is held in its database.

BPL (Broadband over Power): Delivery of broadband data services over traditional electric power lines.

bridge: A device connecting two different network segments at the OSI data-link level.

broadband: A generic term used for the various high bandwidth connection technologies.

browser: The client software developed to retrieve and display Web pages.

bus: A circuit path that connects devices within a computer or over a network.

byte boundary: A subnet mask that “borrows” a full byte of the available host bits.

cache: A dedicated memory area for data or instructions used to speed access to the information.

CBIR (Content-Based Information Retrieval): When programs attempt to use the actual content of the file to perform retrieval instead of metadata associated with it. Examples are image and audio retrieval systems that attempt to match content patterns.

circuit switching: A communication strategy in which direct circuit connections are made between nodes.

client-side: An approach where any needed processing takes place in client software running on a local computer as opposed to on a server.

client-server architecture: Computer communication implemented by software designed to work in tandem where each side has a defined role; client software communicates with the corresponding server software to form requests and receive responses from the server.

Cloud computing: Software and hardware as a service that is delivered to users via the Web.

CMS (Content Management Systems): Any application or platform designed to accommodate creating, editing, and delivering content. In the context of Web publishing, a CMS is a database approach used to create dynamic websites. These include the broader content management framework that has both site building tools as well as ways for programmers and developers to customize the various modules that control the options and features available to them.

color depth: The number of bits used to represent colors in graphics; typically ranges from 8 to 32 bits.

Common Gateway Interface (CGI): Early Web programming interface that handled many types of information exchange between Web applications and servers, such as processing HTML forms. CGI was designed mostly for the UNIX environment and includes programs written in many different languages, including C++, PERL, and Python.

Comp: In the context of Web design, a comp is a graphical representation of a Web page, usually done in a program such as Adobe Photoshop that shows all layout elements, colors, and relative sizing.

compiled program: A program that requires a separate action on the written code that converts it into machine language. The compiled version is an executable file.

computer: In general terms, an electronic, digital device that can accept input and transform it into useful output under the control of a stored program. There are analog computers, and even mechanical ones, but these are the exception.

concatenate: A way to add or combine text strings; it is frequently used in scripting.

content farm (aka a content mill): An enterprise that creates or duplicates Web content solely to maximize potential retrieval through a search engine thereby generating advertising revenue.

cookies: A persistent client-side state object, originally developed by Netscape to compensate for the statelessness of the Web.

CSMA/CD (Carrier Sense, Multiple Access with Collision Detection): The Ethernet technology for managing packet traffic and potential collisions.

CSS (Cascading Style Sheets): A style language supported by the W3C that allows separation of content and presentation in HTML.

daemon: A process running in the background waiting to respond to an event is generically referred to as a *daemon*; a Web server listening at an assigned port is thus referred to as the *HTTPd*.

database management system (DBMS): A program or collection of programs for storing and retrieving information.

dataflow diagrams: A process-centered modeling technique used in systems analysis.

deprecated: In HTML, tags that are no longer supported in the standard.

descriptive markup: Markup that identifies a function or style instead of specific formatting; for instance, the use of the tag for emphasis, but without defining specifically how that will be displayed in every client view.

DHCP (Dynamic Host Configuration Protocol): A protocol that dynamically assigns an IP address to the host for that session; the IP address is “leased” for a period by that host, but once it expires or is released, a different address

might be assigned for future sessions. It is similar to BootP in that it uses an exchange of broadcast messages between the host and some server.

digital: Information that is represented in binary form, as 0s and 1s.

directory tree: The hierarchical structure created on disk drives to organize and collocate files.

dithering: When a nonsupported color is specified in an image but is unavailable to a browser. The browser substitutes a similar available color, often with inconsistent or poor results.

DNS (Domain Name System): A hierarchical system of servers maintaining a database of names and addresses providing these lookup services; this is how IP addresses are associated with a URL.

DNS lookup: What happens with every Web client–server interaction that references a URL; the translation of the URL into a numeric IP necessary for packet addressing.

DOM (Document Object Model): An object-centered view of a page and its elements.

dot notation: The convention that identifies an object within an object hierarchy, such as “document.banner,” or a method, such as “document.write.” In CSS and scripting the dot notation is used for creating classes.

dot pitch: A function of display hardware that refers to the distance between pixels measured in millimeters.

DPI (Dots Per Inch): A measure of display or print resolution.

driver: Software that mediates between an OS and some device or client program.

DSL (Digital Subscriber Line): A digital telephone technology that has most of the advantages of ISDN for lower costs and faster speeds. DSL dedicates part of the available bandwidth to support an analog voice signal, which allows simultaneous computer and voice connection. It is a generic term for a large variety of specific standards that fall into one of two basic types: symmetric, where the upstream and downstream speeds are the same, and asymmetric, where they are different.

DTD (Document Type Definition): In markup languages, a model for a specific class of documents. Part of the SGML specification, it is a language for defining markup elements, their attributes, and their relationships.

EA: An Ethernet address; a 48-bit number that is a specific type of MAC address identifying a node on an Ethernet network.

element: In markup languages, elements are associated with the components of a document or their attributes and are defined by tags.

embedded styles: Style information that is included within an HTML container.

encapsulation: When the data carried by a packet is another type of network packet.

entity relationship (ER) diagrams: A modeling technique used in systems analysis that identifies entities, their attributes, and their relationships. ER diagrams use a standardized set of symbols to represent the components of a database and the cardinality of the relations.

enumerated array: In scripting, an array is a way to store multivalue variables using a two-column table. The left column is a sequence numbered list that comprises the array index.

Ethernet: A data communication network standard that defines a packet type, error controls, and wiring standards. Robert Metcalfe developed the first Ethernet specification at Xerox PARC (Palo Alto Research Center) in 1973. This was revised in 1982 and released as Ethernet II by Digital Equipment Corp, Intel, and Xerox. It was adopted by IEEE (Institute of Electrical and Electronics Engineers) with the release of the 802.3 standard for CSMA/CD networks. Ethernet specifies packets (called frames) as well as wiring standards.

event handlers: In scripting, event handlers are associated with functions or other script commands; they are some event such as a mouse over, mouse click, or a page load that can precipitate an action.

false hits (or drops): In IR, a retrieved item that is deemed nonrelevant; the term comes from early IR systems that used notched cards to represent metadata. The cards representing a document collection could be physically manipulated to segregate a retrieval set in a way that caused potentially relevant records to drop; nonrelevant retrievals were therefore “false drops.”

federated search: In the context of library systems, a metasearch capability that permits a search to be broadcast to the many separate databases within a collection.

flow content: HTML5 implementation of element display equivalent to block-level elements in HTML 4.01.

folksonomy: A hybrid of *folk* and *taxonomy* that are unrestricted user-generated vocabularies resulting in a “group think” approach to describing content.

FQDN (Fully Qualified Domain Name): A name associated with a particular machine or host at a domain.

frequency: The number of cycles per second of a waveform.

frequency modulation (FM): Where a source modulates the frequency of a carrier wave to allow the transmission of information.

FTP (File Transfer Protocol): The TCP/IP protocol developed specifically to upload or download files via an Internet connection.

function: In scripting, also called a *subroutine* or *procedure*, it is a set or block of instructions that execute together to yield a single result; these blocks can be reused in a modular fashion throughout a program.

geolocation: The use of device signals and satellite triangulation to determine the precise geographical location of a device, typically to utilize that information within an API or app.

global variables: In programming, variables with unlimited scope that can be referenced anywhere in a script. PHP has super_global variables that store environment information in arrays.

gopher: Developed at the University of Minnesota, it was a menu-based client-server protocol for access and delivery to information resources.

grids: In cloud computing, it is analogous to the notion of the power grid; it is a collection of cloud computers and applications that can dynamically match needed resources with demand.

hacks: In security, a hack is an attack; in scripting or CSS, it usually refers to an unusual or inelegant solution to some programming problem.

hexadecimal: Base 16 numbering system commonly used to represent large binary numbers, mostly for human convenience. For instance, a 48-bit Ethernet address can be expressed as a 12-digit hexadecimal number. Because 16 digits are needed, the letters a-f are added to the set of 0-9 digits available for base 10 numbers.

homonyms: Refers to words that are either pronounced or spelled the same but have distinct meanings. For instance, a “router” could be a network device or a woodworking tool.

honeypots: Servers that are designed to be easy prey for hackers and used by security experts to study how intruders attack systems and to gather forensic information.

HSL (Hue, Saturation, and Lightness): A color model recognizing that people see hues of color, not mixes of the three additive primary colors. Hue refers to the color's place on the spectrum, represented on a scale of 0–255. Saturation values refer to the clarity of degrees of hue; lightness refers to how light or dark the color appears.

HTML (Hypertext Markup Language): Developed by Tim Berners-Lee, HTML is a text-based standard that defines the tags needed for document structuring and presentation on the Web.

HTTP (Hypertext Transfer Protocol): The protocol that controls the client-server interactions and communication streams for the delivery of content from a Web server to a browser.

hub: A network device that forwards packets to every other device connected to it.

hypertext reference: In HTML, the anchored hypertext reference that creates links within Web pages to other documents or locations.

hypervisor: A software layer that provides a virtual platform on which multiple OS stacks and associated applications can be deployed.

ICMP (Internet Control Message Protocol): Another type of datagram carried in an IP packet just as UDP or TCP packets are; it is used for error checking between gateways and hosts and between different hosts at the Internet layer for error reporting and routing decisions.

image maps: The process of identifying shape coordinate areas in an image and associating them with separate URLs.

include file: An external script file that contains script instructions in a separate file that is included in a source attribute reference instead of being embedded in the HTML within a script tag pair. The different script types are identified by the use of a specific assigned file extension such as .JS or .PHP.

in-degree/out-degree: In link analysis, the number of links pointing to a page, or the number of links to other sources within a particular page.

inline: In HTML, inline elements are rendered on the same line; they do not force a line break. If they do not fit on a line, they will wrap across the screen onto a new line.

instance document: In XML, a document created according to the rules of some specific markup language.

Internet: The global network of computer networks that uses TCP/IP for data communications.

intranets: An internal or local network based on TCP/IP.

inverted file: Also an inverted index or an inverted index file; a structure that results in an index listing items from some field with links to the documents or record where the terms appear and perhaps to their specific location within each document.

inverted index: See inverted file.

invisible Web: By definition any Internet content that is unavailable to search engine spider harvesting is considered “invisible.” It includes content that is dynamically generated, protected by a firewall or the robot exclusion protocol, or within file formats harvesting programs cannot process.

IP address: The logical address associated with some Internet node using either IPv4 or IPv6. Address classes have been established, and organizations are assigned a class in the domain registration process. The IP address space assigned to an organization is handled internally by associating device MAC addresses with available IP addresses or through dynamic IP address assignment.

IP packets: The packets specified by TCP/IP that move data from one host to another.

IR (Information Retrieval): As described by Korfage (1997, p. 324), IR is “the location and presentation to a user of information relevant to an information need as expressed by a query.” Essentially, IR systems function as an intermediary between information resources and a user’s information need.

IRC (Internet Relay Chat): A real-time Internet connection used for chat sessions.

ISPs (Internet Service Providers): The companies that provide access to Internet services for the public through dialup or broadband options.

JavaScript: A lightweight, client-side scripting programming language that was created in 1995 at Netscape, Inc. Commonly used in Web development in conjunction with HTML. It shares some feature with Java, but Java is a full featured, object-orientation programming language.

JQuery (jquery.com): A JavaScript code library that allows a complex script to be added to a page by referencing a library script name and location.

JSON (JavaScript Object Notation): A data exchange format based on JavaScript used as an alternative to XML for exchanging data among Web applications.

JVM: The Java Virtual Machine, a cross platform layer for running Java applications.

LDAP (Lightweight Directory Access Protocol): An IP protocol that facilitates data exchange between programs to gather data from various network directories of users or devices. It is both a network protocol and architecture for organizing directory data.

link bomb: The deliberate manipulation of the Google Page Rank algorithm by making links to specific phrases to cause unusual retrieval results for a search.

logical markup: See descriptive markup.

loopback address: The IP address 127.0.0.1 that is used for testing TCP/IP installations.

lossy compression: A file format that can have data loss with the compression algorithm; JPG is an example of such a format.

MAC (Media Access Control) addresses: A hardware identifier for a network device; for Ethernet networks, this is the Ethernet address. MAC addresses are mapped to IP addresses for Internet hosts.

macro viruses: Viruses written in the Visual Basic scripting language used by Microsoft for application macros. VB Macros make it possible for data files to carry a virus.

malware: A generic term used to describe any of the programs designed to annoy, damage, or otherwise invade your computer and includes viruses, worms, and Trojan horses as well as various types of spyware and adware.

markup languages: Text-based languages that use special or reserved text characters to convey structural or presentational information to a parsing program.

mashups: Hybrid functionalities made possible through shared APIs, such as the use of the Google Maps API to add map functions to a Website.

metasearch: A simultaneous search of multiple databases or search engines.

microdata: A specific type of semantic metadata that can be embedded within HTML in the form of added attributes.

microservice architecture: A modular approach to building complex systems by connecting together separate programs, APIs, and other services to handle needed functions as opposed to a single monolithic integrated system.

MIME (Multipurpose Internet Mail Extensions): The automated encoding scheme that facilitates the transfer of binary information in an ASCII text representation; for instance, MIME enables binary file email attachments.

mobile malicious code: Any program that moves from computer to computer via a network that is intentionally designed to modify a computer system without the user's consent.

modem: A modulator/demodulator, a device that allows a computer to connect to other computers via an analog telephone line.

MOOC (Massive Open Online Classes): Online Web-delivered classes that are available to anyone with few or no restrictions on the number or status of participants; often associated with private companies and/or colleges and universities.

Multiplexing/demultiplexing: Describes networks where packets can be mixed together with other unrelated packets (*multiplexing*); when they are separated from each other on the receiving end, it is called *demultiplexing*.

MUVE (Multiuser Virtual Reality Environment): A virtual world, such as Second Life, that is shared by multiple simultaneous users.

namespace: In XML, a way to define a vocabulary that can then be shared with other markup languages.

NAT (Network Address Translation) Protocol: A way to manage the correspondence between public and private network addresses; it is like a proxy server acting as an intermediary for the packet streams.

net neutrality: A policy issue that determines how packets are treated; with net neutrality, all packets are equal, and no packets are given "priority" status on the Internet over others.

NetBIOS (Network Basic Input/Output System): Network ports used for functions such as file and printer sharing.

node: In networks, each connected device is a node. In XML, the hierarchical tag structure is also referred to as a node tree, and each element is a node.

normalization: A formal process based on relational database theory to test data structures to ensure the database integrity.

OAIS (Open Archival Information System): A specialized form of CMS for submitting and managing digital content and its associated metadata for the purpose of its long-term curation and delivery to users in appropriate Web formats.

octal values: In the context of UNIX file permissions, octal values are a short-hand way to refer to the eight different combinations of read, write, and execute permission that are possible.

octet: A byte value in an IP address; 8 bits of the 32 bits used in IPv4.

open source: Community-based software development that is free to download and adopt. Developers have access to the source code and are free to extend the software and support other users.

P2P (peer-to-peer): A direct connection between two computers in which there is no central server and little needed hardware; for example, a direct connection between the two computer Ethernet ports with a crossover cable would form a P2P network. In the context of the Internet, it refers to file-sharing networks enabled by client software such as Gnutella to facilitate data exchange between nodes.

packet encapsulation: When the datagram of a packet is another packet; packets nested within packets. For instance, an IP packet typically carries a TCP or UDP packet in its data segment.

packet switching: A data communication strategy where information is broken up into small, discrete packets with a source and a destination addresses. The packets are transmitted over a network along with lots of other unrelated packets headed toward their destination.

packets: The datagrams that result when data is broken up into discrete chunks, labeled and addressed, and sent out over a network.

PageRank: The algorithm developed by Google founders Sergey Brin and Larry Page that uses link analysis to contribute to a relevance measure.

parameter: Something that a command or function uses, stores, or acts on; parameters can be passed to commands or functions.

PC: A personal computer. Often associated with Windows, it generically includes other platforms such as Apple and Linux.

PCIe (Peripheral Component Architecture): A point-to-point computer data bus that moves a serial stream of 8-bit data packets.

Peer-to-peer: A network connection where no centralized server controls the data communication.

performance-based design: Design techniques that require rethinking how all elements will perform when used, especially as applied to the mobile Web. Performance-based solutions include using CSS for graphic effects.

pharming: A technique that guides or redirects users to fake or counterfeit sites that mimic a legitimate company or present the appearance of some official government site to fool users and collect personal or financial information from them.

phishing: A form of “social engineering,” which refers to any technique designed to trick individuals into revealing personal or other information with

fake emails designed to trick users and entice them to volunteer personal information.

PHP: A scripting language that grew out of an earlier language developed by Rasmus Lerdorf in 1995 called “personal home page tools” but now refers to the recursive name PHP hypertext preprocessor. It is a server-side language, and thus requires a server PHP engine to preprocess scripts before their output is sent the client. It is frequently used in conjunction with SQL databases for dynamic websites and CMS.

phrasing content: HTML5’s handling of HTML 4.01 inline elements.

physical markup: See procedural markup.

ping: A utility run from a command prompt on Internet-connected hosts; it forms a message based on an ICMP echo request directed to a host to see if it is “alive.”

pipelining: In HTTP 1.1, a method that permits multiple requests to be processed simultaneously.

pixel: A contraction of the two words *picture* and *element* and abbreviated as px; it refers to a single dot on a display, the number of which is determined by the resolution setting.

polysemy: Refers to one word that may have related but quite distinct meanings depending on the context, such as the term *mouth*, which could be a body part or the end of a river.

port: A communication channel; it might be associated with a physical device, such as a USB or parallel port for a printer, or a logical port, such as the assignments made by TCP/IP.

portal: In the context of Web design, a portal is considered a site that acts as the primary entry point for the Internet activities of users.

PPP-over-Ethernet: A newer variation of PPP that is used when the DSL modem acts as a bridge or switch rather than a simple router.

precision: In IR, precision is a performance measure that is based on the proportion of relevant retrievals within a search set.

probabilistic: One of the classic IR models that attempts to algorithmically estimate the probability that a retrieval is relevant to a query.

procedural markup: In HTML, markup that describes the exact appearance or function of an element, such as using the bold tag to add formatting to text.

protocol: The set of rules for device and data communication.

proxy server: A host that stands in as a proxy for another host’s Internet activities; it serves as an intermediary for the packet streams. They are often used to authenticate users to a service and to restrict or filter Internet access.

raster graphic: A format where shapes are approximated by filling in squares within a grid.

RDF (Resource Description Framework): One of the key components of the W3C Semantic Web initiative, it is an XML-based application for resource description based on identifying essential resource properties and storing each in the form of a property type and value pair.

RDFa (RDF with attributes): A set of attributes for adding semantic metadata that is directly embedded within HTML5, XHTML, and XML content that provides machine-readable metadata in the form of property-value pairs.

RDF triple: A semantically meaningful RDF statement comprised of a subject, a predicate, and an object.

recall: A performance measure that is based on the number of relevant retrievals in a set compared to the total number of relevant documents in some corpus. Determining that total is problematic outside of IR test bed environments.

relational database: A database designed and implemented according to the rules of the relational model where the data and data relationships are stored in a set of interconnected normalized tables. Used by many CMSs for storing data.

relative reference: In HTML, referencing some other resource on the same server within an anchor or image tag by providing only the information needed to locate that file relative to the location of the referring file in the directory tree.

resolution: There are various contexts for this term—*monitor resolution* refers to two different measures: the vertical resolution is the number of rows of pixels on the screen, and the horizontal resolution is the number of intensity changes permitted across each row of pixels. Therefore, the vertical resolution is the number of pixels in a screen column, and the horizontal is the number of pixels in a row across the screen. *Bit resolution* is the number of bits associated with each pixel. Print resolution is DPI setting for the output.

responsive design: Design approach that seeks to create a site that works well across many platforms and devices. As described by Ethan Marcotte who coined the term, responsive design employs three technical elements: fluid grids, flexible images, and CSS3 media queries.

root element: In markup languages, the root element is the top-level element that by definition has no parent element; it is a starting tag not nested within any other tag. A defined root element is required by a DTD.

rootkit: Malware that gets embedded deep in the system area of a computer and can result in the system being compromised by an intruder or hijacked to become a spambot or worse.

Router: A device designed to forward packets between networks or segments. Routers use IP addresses and a subnet mask to determine whether the packets are destined for a location within the network or for some outside Internet destination.

RSS (Really Simple Syndication or alternatively, Rich Site Summary, or RDF Site Summary): An XML-based protocol that enables a variety of syndicated feeds of blogs or podcasts.

schema: In the context of database systems, a schema is developed from the process of system analysis and the modeling of the database; the schema reflects the structure of a database and helps visualize or describe its organization. Schema are also used in XML.

scripting languages: A lightweight form of programming where textual commands are interpreted by some OS or parsing program.

segments: Subsets of the nodes managed as a group within the larger network.

server-side: Web scripts are divided into two different types based on where they are executed. *Client-side* scripts are executed solely on the client host; *server-side* scripts execute on a server with a processing engine or container installed and configured to coordinate with a Web server.

servlet: A Java application that runs on a server in an assigned servlet container.

SGML (Standard Generalized Markup Language): A meta-language derived from GML that specifies how to create markup languages. Developed as an ANSI standard in 1983, it became an ISO standard in 1986 and is the foundation of HTML.

SIP (Session Initialization Protocol): A form of instant communication over a client–server Internet connection. In the context of OAIS, SIP is also for “submission information package.”

SLIP/PPP (Serial Line Internet Protocol/Point-to-Point Protocol): Protocols that assign an IP address to a host via a dial-up connection. SLIP is an older standard that does not have the compressions and error checking that PPP does.

SMTP (Simple Mail Transfer Protocol): The TCP/IP client–server protocol that supports email.

SNMP (Simple Network Management Protocol): A protocol that automates the process of managing devices such as routers, computers, printers, or servers on an IP network using agent software running on the device to store data in a database called the Management Information Base. When requested, the agent supplies that node-specific data to another program, the Network Management System that is run on a manager host.

SOAP (Simple Object Access Protocol): W3C describes SOAP as an XML-based mechanism for exchanging information in a decentralized and distributed environment.

spam: The generic term for unwanted and unsolicited mail. The term is from the classic Monty Python restaurant skit where all the breakfast options included

varying amounts of the processed meat known as SPAM™, whether the customer wanted it or not.

splash page: An initial website page designed to catch the attention of a visitor.

sprite file: A single file where different parts of one image appear with shifting background positions, using fonts for logos, and scalable vector graphics; a CSS approach to create flexible images.

spyware: Any malware that uses techniques such as bots, flash cookies, key loggers, or third-party cookies to track your activities and gather personal information from your computer, usually to sell or use the information for targeted advertising. However, these programs can be potentially much more dangerous as they can capture keystrokes to gather passwords or credit card numbers.

SQL (Structured Query Language): A standardized command-based language for querying relational databases.

SSID (Service Set Identifier): A network identifier that provides a minimum level of access control to a WLAN but is not highly secure because it is not encrypted and is often automatically assigned.

statelessness: The lack of an ongoing stateful connection between client and server. The HTTP protocol was developed for single communication events between client and server ending with the closing of the connection; because no record of the transaction is kept by the server the condition known as *statelessness* results. With a stateless protocol, there is just one transaction per connection, and multiple separate connections are needed to retrieve a page if multiple files such as images are associated with it.

stickiness: The degree to which a website is designed to retain users.

stopwords: In document processing, words in documents that are deemed meaningless as index terms and removed, such as articles, prepositions, and conjunctions.

streaming: A client server strategy for delivering multimedia over the Internet that delivers content at a data rate that matches the available bandwidth, allowing continuous playback.

subnet: A network management strategy that separates nodes into separate groups, which are created by “borrowing” some of the bits reserved for host ID to extend the network ID portion.

subnet masks: Generically, a mask refers to the idea of hiding something to isolate one element of interest; a subnet mask is a number used to isolate the network or host portions of packet addresses to enable routing decisions.

subroutine: See function.

switch: A network device similar in function to a hub that serves as a network connection point for many devices. Switches use the MAC addresses associated with the frames (Ethernet packets) and filter the packet streams,

450 Glossary

but unlike hubs, switches forward packets only to the device with the correct address.

synonymy: Means that many terms can represent the same concept; synonyms are different words with similar meanings (e.g., automobile or car).

tags: In the context of markup languages, tags are used to identify elements by using reserved text. In HTML and XML, tags are identified by the use of the less than and greater than signs.

TCP/IP (Transmission Control Protocol/Internet Protocol): The packet-switching technology used to support all Internet activities.

telnet: An Internet protocol that facilitates a real-time connection between two Internet hosts.

term stemming: In text processing, stemming refers to truncating a term to its root.

term weights: In text processing, weights can be assigned to terms in addition to simply tracking the presence or absence of a term. Term frequency, position, field location, or decoration can all be incorporated into weight calculation.

topology: The physical layout of the wired connections in a network. Common topologies are:

bus: Where each node comes off the connecting backbone wire in a linear series.

mesh: Where every node is connected to every other node, ensuring direct communication between every pair of devices.

star: Where wires to nodes radiate out from a central server.

token ring: Where computers are connected together in a circular arrangement, and a special packet, called the *token*, moves around to each node and determines what device can send data.

tree: A hybrid of a bus and a star, with potentially multiple stars coming off a central bus.

Trojan horse: As the name implies, Trojan horse programs are imposters; they claim to be something good but are, in fact, malicious. They depend on misleading the user to run them, and this misrepresentation is their defining characteristic. Unlike worms, they are nonreplicating, and unlike viruses, they do not infect other files. Nevertheless, if they succeed in tricking the user to run them, they can serve as an entry point for a future attack.

typography: How text characters appear when printed on a page or screen; typography is expressed as different fonts and sizes.

UDP (User Datagram Protocol): A packet type used to carry IP data when reliable delivery is not a priority. UDP is an alternative packet type that does not have the built-in error checking and acknowledgement features of TCP that

ensure the integrity of the delivered data. UDP is primarily used for streaming protocols where speed is prioritized and packet delivery acknowledgement is unnecessary.

URI (Universal Resource Identifier): The unique identifier for a point of contact on the Internet.

URL (Uniform Resource Locator): The standard Web page locator as approved by the IETF; it includes a protocol, FQDN, and filename and location on a Web server.

UTP (Unshielded Twisted Pair): Inexpensive copper wiring used for phone lines and Ethernet connections.

UXD (User experience design): The application of usernomics from human factors engineering to the user experience; in Web design, it emphasizes the centrality of the user in all aspects of the design process.

variable: Generically, a placeholder for some value provided or assigned later. In programming, variables are associated with memory locations.

vector images: Graphics where shapes are determined by applying mathematical formulas instead of filling in squares on a grid.

vector space model: An information retrieval model that attempts compare the angles between document and query term vectors and calculate a similarity measure between them to algorithmically assess the relevance of a document to a query.

VERONICA (Very Easy Rodent-Oriented Net-wide Index to Computer Archives): An early automatically generated searchable index for Internet Gopher resources; only the terms in the menu items were indexed—the full text was not.

viewport: A generic term for any device used to display content.

virtual machine: A software implementation to create a virtual computer that behaves just as a “real-world” computer would, with an OS that can run applications. A portion of a system is carved out to provide each full instance of often multiple machines that may emulate different architectures and run separate from each other for each user.

virtualization: In computing, the process of virtualizing hardware, software, or networks by emulating the “real world” version of something that will in turn behave in the same way as the physical device, software, or service.

virus: A form of malicious mobile code; executable code that acts either as a file infector or a boot sector infector.

VoIP: Voice over IP is a TCP/IP alternative to POTS for telephone service.

VPN (Virtual Private Network): A strategy to extend an address space by the creation of a virtual, secure “tunnel” through which authentication data and private network traffic can pass, allowing a remote user to access their organization’s network from the outside.

WAN (Wide Area Network): A collection of local area networks.

WCM (Web Content Management): A CMS specifically for managing and delivering Web content. WCMs depend on databases, scripting, and styles to create dynamic websites.

Web (World Wide Web): The part of the Internet utilizing the HTTP protocol and results in a global hypertext database of information resources, many of which are in HTML format. Developed by programmer Tim Berners-Lee at CERN, it was first publicly demonstrated during the Hypertext '91 Conference in San Antonio, Texas.

Web 2.0: Coined by Tim O'Reilly of O'Reilly Media, Web 2.0 is the view of the Web as a platform, delivering useful applications and enabling a more dynamic, interactive, and participatory Web.

Web 3.0: Viewed as the evolution of the Web into an intelligent, semantic Web that makes extensive use of XML and metadata to facilitate program-to-program communication as well as enhanced intellectual access to resources.

Web bug: A very small or transparent graphic embedded in a Web page that is associated with an HTTP set-cookie request by a third-party server; often an ad-server but could also be associated with hackers collecting information to be used in some potential future attack.

Wi-Fi (wireless fidelity): An IEEE standard for wireless connectivity by way of a broadcast signal using WAP (Wireless Application Protocol).

wiki: A collaborative software tool, developed by Ward Cunningham. The name is from the Hawaiian for “quick” and refers to software enabling anyone to add or edit Web content without having to know HTML or any details about the hosting platform. Wikis have become an important source of Internet-based reference, such as the online encyclopedia at Wikipedia.com, but they also present opportunities to build and share collaborative content in other contexts, such as internal knowledge bases or other collaborative projects.

wireframe: A stripped down visual representation of a website to show and model the site architecture. Wireframe schematics can show the overall layout, a content inventory, and various page elements such as the location of headers, footers, or images on pages.

World Wide Web Consortium (W3C): The coordinating standards body that oversees Web development.

worm: Often lumped together with Trojans, they are somewhat different in action. Worms can turn computers into “zombie” spambots and cause thousands of replicated copies of itself to be sent out over a network and take part in “denial of service” attacks. They can infect local files and cause the afflicted host to lose data or crash.

WWW: See Web.

WYSIWYG (What-you-see-is-what-you-get): When a program displays a final presentation view of a document; for instance, an HTML editor that instead of displaying code, shows how a browser would render the document.

XHTML: HTML that is compliant with the more strictly enforced rules of XML.

XML (Extensible Markup Language): A lightweight meta-language derived from the SGML standard that has been optimized for the Web.

This page intentionally left blank

Bibliography

- A new wave. (2002). *The Economist*, 362(8256), 68.
- About Drupal. (2013). Retrieved July 15, 2013, from <https://drupal.org/about>.
- About Kaltura. (2012). Retrieved July 17, 2013, from <http://corp.kaltura.com/>
About-Kaltura.
- About microformats. (n.d.). Retrieved January 27, 2014, from <http://microformats.org/about>.
- About rich snippets and structured data. (2014). Retrieved January 5, 2014, from <https://support.google.com/webmasters/answer/99170?hl=en>.
- Abram, S. (2005). The Google opportunity. *Library Journal*, 130(2).
- Abram, Stephen. (2009, September 9). APIs and libraries. Retrieved August 30, 2013, from <http://stephenslighthouse.com/2009/09/01/apis-and-libraries/>
- Adamo, Stephanie. (2013, January 11). comScore releases December 2012 U.S. search engine rankings. Retrieved July 22, 2013, from http://www.comscore.com/Insights/Press_Releases/2013/1/comScore_Releases_December_2012_U.S._Search_Engine_Rankings.
- Adams, Douglas. (2002). *The salmon of doubt: Hitchhiking the galaxy one last time* (1st ed.). New York: Harmony Books.
- Adams, Shar. (2013, June 4). The business of Google Fiber. Retrieved June 4, 2013, from <http://www.theepochtimes.com/n3/89237-the-business-of-google-fiber/>
- Albanesius, Chloe. (2012, April 24). 20 Percent of Macs infected with Windows malware. Retrieved September 28, 2013, from <http://www.pcmag.com/article2/0,2817,2403463,00.asp>.
- Allen, M. (2008). Web 2.0: An argument against convergence. *First Monday*, 13(3).
- Ally, Mohamed, & Needham, Gill. (2012). *M-libraries: Transforming libraries with mobile technology*. London: Facet Publishing.
- Alpcan, T., Bauckhage, C., & Kotsovios, E. (2007). Towards 3D Internet: Why, what, and how? Retrieved August 28, 2013, from http://www.kotsovios.com/research/papers/Alpcan-3D_Internet.pdf.
- American Library Association (ALA) Research Series. (2007). Connect communities: Public library funding & technology Access study 2006–2007 Report.

456 Bibliography

- Chicago: American Library Association. Retrieved May 1, 2008 from <http://www.ala.org/ala/ors/plftas/finalreport.pdf>.
- American Library Association's (ALA) Presidential Committee on Information Literacy. (2006, July 24). Final report. Retrieved May 18, 2008, from <http://www.ala.org/ala/acrl/acrlpubs/whitepapers/presidential.cfm>.
- Amerland, D. (2014). *Google semantic search: Search engine optimization (SEO) techniques that gets your company more traffic, increases brand impact and amplifies your online presence* (1st ed.). Indianapolis, IN: Que.
- Anderson, C. (2004, October). The long tail. *Wired*, 12, 170–177.
- Anderson, Chris, & Wolff, Michael. (2010, September). The Web is dead. Long live the Internet. *Wired*, 11.
- Anderson, Nate. (2013, March 24). How I became a password cracker. Retrieved May 1, 2013, from <http://arstechnica.com/security/2013/03/how-i-became-a-password-cracker/2/>
- Apple iPad launch marred by technical problems. (2010). Retrieved April 11, 2013, from <http://www.telegraph.co.uk/technology/apple/7558319/Apple-iPad-launch-marred-by-technical-problems.html>.
- Apple launches iPad 2. (2011, March 2). Press release. Retrieved May 21, 2011, from <http://www.apple.com/pr/library/2011/03/02Apple-Launches-iPad-2.html>.
- Arnold, B. (2008). Sizing the web: Domains, sites, hosts. Retrieved May 1, 2008, from <http://www.caslon.com.au/metricsguide1.htm#domains>.
- Associated Press. (2006, January 20). Google won't hand over files. Retrieved October 1, 2007, from <http://www.wired.com/politics/law/news/2006/01/70055>.
- Azad, K. (2007, January 11). Using JSON to exchange data. Retrieved December 9, 2013, from <http://betterexplained.com/articles/using-json-to-exchange-data/>
- Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*. New York: ACM Press.
- Baeza-Yates, R., & Ribeiro-Neto, B. (2011). *Modern information retrieval: The concepts and technology behind search*. New York: Addison Wesley.
- Barroso, L. A., Dean, J., & Hölzle, U. (2003). Web search for a planet: The Google cluster architecture. *Institute of Electrical and Electronics Engineers (IEEE) Micro*, 23(2), 22–28.
- Bates, M. J. (2002). *Toward an integrated model of information seeking and searching (Keynote)*. Paper presented at the Fourth International Conference on Information Needs, Seeking and Use in Different Contexts, Lisbon, Portugal.
- Battelle, J. (2005). *The search*. New York: Penguin Group.
- Bawden, D. (2007). Information seeking and retrieval. *ALISE*, 28(2), 126.
- Baxter, G., & Anderson, D. (1996). Image indexing and retrieval: Some problems and proposed solutions. *Internet Research*, 6(4).
- BBC News. (2003, December 7). "Miserable failure" links to Bush. Retrieved May 10, 2008, from <http://news.bbc.co.uk/2/hi/americas/3298443.stm>.
- BBC News. (2007, May 14). Web 2.0 "neglecting good design." Retrieved July 1, 2013, from <http://news.bbc.co.uk/2/hi/technology/6653119.stm>.
- Bell, Killian. (2011, November 9). Adobe announces the end of Flash player for mobile devices. Retrieved July 16, 2013, from <http://www.technobuffalo.com/2011/11/09/adobe-announces-the-end-of-flash-player-for-mobile-devices/>
- Bell, L., Pope, K., Peters, T., & Galik, B. (2007). Who's on third in Second Life? *Online*, 31(4).
- Bergman, M. K. (2001, September 24). The "deep" Web: Surfacing hidden value. Retrieved October 1, 2007, from <http://www.brightplanet.com/images/stories/pdf/deepwebwhitepaper.pdf>.

- Berinstein, P. (1999). Do you see what I see? Image indexing for the rest of us. *Online*, 23(2), 85–88.
- Bernard, M., Mills, M., Peterson, M., & Storrer, K. (2001). A comparison of popular online fonts: Which is best and when? *Usability News*, 3(2).
- Berners-Lee, T. (1999). *Weaving the Web: The original design and ultimate destiny of the World Wide Web by its inventor*. New York: HarperCollins.
- Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic Web. *Scientific American*, 284(5), 10.
- Berlot, J. C., McClure, C. R., Wright, C. B., Jensen, E., & Thomas, S. (2009). Public libraries and the Internet 2009: Study results and findings. Retrieved May 14, 2014, from http://www.ii.fsu.edu/content/download/16874/109694/03_executive%20summary1-7.pdf.
- Bhola, J. (2002). *Wireless LANs demystified*. New York: McGraw-Hill.
- Big data. (2013). Retrieved September 10, 2013, from <http://www.gartner.com/it-glossary/big-data/>
- Bishop, T. (2004, January 26). Microsoft notebook: Wiki pioneer planted the seed and watched it grow. Retrieved October 2007, from http://seattlepi.nwsource.com/business/158020_msftnotebook26.html.
- Black, P. E. (2006). Dictionary of algorithms and data structures [online]. Retrieved September 13, 2008, from <http://www.nist.gov/dads/HTML/invertedIndex.html>.
- Black, P. E. (2008). Dictionary of algorithms and data structures [online]. Retrieved September 13, 2008, from <http://www.nist.gov/dads/HTML/invertedIndex.html>.
- Black, U. (1999). *Advanced Internet technologies*. Upper Saddle River, NJ: Prentice Hall.
- Bort, Julie. (2013, May 13). Nearly 500 million searches a day are for things Google has never heard of. Retrieved July 19, 2013, from <http://www.businessinsider.com/500m-things-google-has-never-heard-of-2013-5>.
- Bosak, J., & Bray, T. (1999). XML and the second-generation Web. *Scientific American*, 280(5), 89–94.
- Boutell.com. (2007, February 15). How many websites are there? Retrieved October 1, 2007, from <http://www.boutell.com/newfaq/misc/sizeofweb.html>.
- Brault, Matthew W. (2012, July). Americans with disabilities: 2010: Household economic studies. Retrieved July 1, 2013, from <http://www.census.gov/prod/2012pubs/p70-131.pdf>.
- Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., Yergeau, F., & Cowan, J. (2006). Extensible markup language (XML) 1.1 (2nd ed.). Retrieved May 1, 2008, from <http://www.w3.org/TR/xml11/#sec-xml11>.
- Breeding, M. (2007). Next-generation library catalogs. *Library Technology Reports*, 43(4), 5–42.
- Breeding, M. (2014). Library Systems Report 2014. *American Libraries*, 45(5), 21–33.
- Brenner, Joanna. (2013, February 14). Pew Internet: Social networking. Retrieved May 20, 2013, from <http://pewinternet.org/Commentary/2012/March/Pew-Internet-Social-Networking-full-detail.aspx>.
- Brenner, Joanna, & Rainie, Lee. (2012, December 9). Pew Internet: Broadband. Retrieved June 3, 2013, from <http://pewinternet.org/Commentary/2012/May/Pew-Internet-Broadband.aspx>.
- Brenner, Joanna, & Smith, Aaron. (2013, August 5). 72% of Online adults are social networking site users. Retrieved September 5, 2013, from <http://pewinternet.org/Reports/2013/social-networking-sites.aspx>.
- Brenner, S. E., & Aoki, E. (1996). *Introduction to CGI/PERL*. New York: M&T Books.

458 Bibliography

- Bright, Peter. (2011a, March 22). How operation b107 decapitated the Rustock botnet. Retrieved June 11, 2013, from <http://arstechnica.com/information-technology/2011/03/how-operation-b107-decapitated-the-rustock-botnet/>
- Bright, Peter. (2011b, March 29). Rustock repercussions: Spam down by a third, at least for now. Retrieved June 11, 2013, from <http://arstechnica.com/security/2011/03/rustock-repercussions-spam-down-by-a-third-at-least-for-now/>
- Brin, S., & Page, L. (1998). *The anatomy of a large-scale search hypertextual Web search engine*. Paper presented at the Proceedings of Seventh World Wide Web Conference, Brisbane, Australia.
- Broder, A., Fontoura, M., Josifovski, V., & Riedel, L. (2007). A semantic approach to contextual advertising. *ACM SIGIR International Conference on Research and Development in Information Retrieval*, 559–566. New York, NY: ACM Press.
- Brodkin, Jon. (2013, June 3). The secret to online safety: Lies, random characters, and a password manager. Retrieved June 19, 2013, from <http://arstechnica.com/information-technology/2013/06/the-secret-to-online-safety-lies-random-characters-and-a-password-manager/>
- Buckley, Thomas, & Radicati, Sara. (2012, October 12). Email market, 2012–2016. Retrieved June 11, 2013, from <http://www.radicati.com/wp/wp-content/uploads/2012/10>Email-Market-2012-2016-Executive-Summary.pdf>.
- Bugeja, M. J. (2007). Second thoughts about Second Life. *Chronicle of Higher Education*, 54(3), C2–C4.
- Bush, V. (1945). As we may think. *Atlantic Monthly*, 176, 101–108.
- Byron, Angela, Berry, Addison, & De Bondt, Bruno. (2012). *Using Drupal* (2nd ed.). Sebastopol, CA: O'Reilly.
- BytePile.com. (2002, October 19). DSL categories. Retrieved August 31, 2007, from http://www.bytepile.com/dsl_categories.php.
- Cafaro, Massimo, & Aloiso, Giovanni. (2010). *Grids, clouds and virtualization* (1st ed.). New York: Springer.
- Calhoun, K. (2006, March 17). The changing nature of the catalog and its integration with other discovery tools. Retrieved November 18, 2013, from <http://www.loc.gov/catdir/calhoun-report-final.pdf>.
- CalTech Information Technology Services. (2003, January 6). NetBIOS blocked at campus border. Retrieved August 1, 2003, from <http://www.its.caltech.edu/its/security/policies/netbios-block.shtml>.
- Canavan, J. (2007, February 13). W32.Swen.A@mm. Retrieved June 19, 2013, from http://www.symantec.com/security_response/writeup.jsp?docid=2003-091812-2709-99.
- Cannarella, J., & Spechler, J. A. (2014, January 17). Epidemiological modeling of online social network dynamics. Retrieved January 22, 2014, from <http://arxiv.org/pdf/1401.4208v1.pdf>.
- Carr, N. (2008a). *The big switch*. New York: W.W. Norton & Company.
- Carr, N. (2008b, July/August). Is Google making us stupid? *Atlantic Monthly*, 302(1), 56.
- Case, D. O. (2007). *Looking for information: A survey of research on information seeking, needs, and behavior* (2nd ed.). New York: Academic Press.
- Casey, M. E., & Savastinuk, L. C. (2007). *Library 2.0: A guide to participatory library service*. Medford, NJ: Information Today.
- Castelli, V. (Ed.). (2002). *Encyclopedia of library and information science* (Vol. 71). New York: Marcel Dekker, Inc.
- CERN. (2008). Tim Berners-Lee's proposal. Retrieved May 8, 2014, from <http://info.cern.ch/Proposal.html>.

- Chacos, B. (2014, January 14). Appeals court strikes down FCC's net neutrality rules. Retrieved January 14, 2014, from <http://www.pcworld.com/article/2087441/appeals-court-strikes-down-fccs-net-neutrality-rules.html>.
- Chakrabarti, S., Dom, B., Kumar, S.R., Raghavan, P., Rajagopalan, S., Tomkins, A., et al. (1999). Hypersearching the Web. *Scientific American*, 280(6), 54–61.
- Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Gruber, R.E. (2006). Bigtable: A distributed storage system for structured data. Retrieved December 14, 2013, from <http://static.googleusercontent.com/media/research.google.com/en/us/archive/bigtable-osdi06.pdf>.
- Chant, I. (2014). Research: Pew Concludes Library Series. *Library Journal*, 139(8), 19.
- Charles, G.T., Jr. (1997). *LAN blueprints*. New York: McGraw-Hill.
- Cheng, H.-L., & Rasmussen, E.M. (1999). Intellectual access to images. *Library Trends*, 48(2), 291–302.
- Cheng, Jacqui. (2013, March 21). Apple follows Google, Facebook, and others with two-step authentication. Retrieved June 19, 2013, from <http://arstechnica.com/apple/2013/03/apple-follows-google-facebook-and-others-with-two-step-authentication/>
- Cheong, F.C. (1996). *Internet agents: Spiders, wanderers, brokers, and bots*. Indianapolis, IN: New Riders Publishing.
- Chien, E. (2007, February 13). Nimda mass mailing worm. Retrieved June 9, 2013, from http://www.symantec.com/security_response/writeup.jsp?doc_id=2001-091816-3508-99.
- Chien, E. (2014, April 14). Heartbleed poses risk to clients and the Internet of things. Retrieved May 10, 2014, from <http://www.symantec.com/connect/blogs/heartbleed-poses-risk-clients-and-internet-things>.
- Christensen, C.M. (2003). *The innovator's dilemma: The revolutionary book that will change the way you do business*. New York: Collins.
- Chu, Heting. (2003). *Information representation and retrieval in the digital age*. Medford, NJ: Information Today.
- Chu, Heting. (2010). *Information representation and retrieval in the digital age* (2nd ed.). Medford, NJ: Information Today.
- Ciocco, R., & Huff, A. (2007). Mission IM-possible: Starting an instant message reference service using Trillian. *Computers in Libraries*, 27(1), 26–31.
- Cisco Systems. (2008). Simple network management protocol (SNMP). In *Internetworking Technology Handbook*. Retrieved May 1, 2008, from <http://www.cisco.com/en/US/docs/internetworking/technology/handbook/SNMP.html>.
- Clark, Jason A. (2012). *Building mobile library applications*. Chicago, IL: Library and Information Technology Association.
- CNET Wireless Resource Center. (2013a). Wi-Fi. Retrieved July 18, 2013, from http://www.cnet.com/1990-7363_1-6361076-3.html.
- CNET Wireless Resource Center. (2013b). 3G. Retrieved July 18, 2013, from http://www.cnet.com/1990-7363_1-6361076-4.html.
- Coffeen, Tom. (2013, April 4). A brief snapshot of IPv6 adoption. Retrieved June 7, 2013, from <http://www.infoblox.com/community/blog/brief-snapshot-ipv6-adoption>.
- Coleman, J., Katz, E., & Menzel, H. (1957). The diffusion of an innovation among physicians. *Sociometry*, 20(4), 253–270.
- Comer, D.E., & Stephens, D.L. (1991a). *Internetworking with TCP/IP Volume I: Principles, protocols, and architecture* (2nd ed.). Englewood Cliffs, NJ: Prentice Hall.
- Comer, D.E., & Stevens, D.L. (1991b). *Internetworking with TCP/IP Vol II: Design, implementation, and internals*. Englewood Cliffs, NJ: Prentice Hall.

460 Bibliography

- Compare content management systems. (2013). Retrieved July 8, 2013, from <http://www.cmsmatrix.org/>
- Coombs, Karen A., & Hollister, Amanda J. (2010). *Open source Web applications for libraries*. Medford, NJ: Information Today, Inc.
- Costello, Sam. (2013). How many apps are in the iPhone app store. Retrieved May 20, 2013, from <http://ipod.about.com/od/iphonesoftwareterms/qt/apps-in-app-store.htm>.
- Cote, Denise, Kraemer, Beth, Nahl, Diane, & Ashford, Robin. (2012). Academic librarians in Second Life. Retrieved August 26, 2013, from http://uknowledge.uky.edu/libraries_facpub/17.
- Courtney, Nancy. (2007). *Library 2.0 and beyond: Innovative technologies for tomorrow's user*. Westport, CT: Libraries Unlimited.
- Coventry, M. (2006). Libraries for a new generation: Getting the net gen on the right information highway. *UMNNews*. Retrieved October 1, 2007, from http://www1.umn.edu/umnnews/Feature_Stories/Libraries_for_a_new_generation.html.
- Coyier, Chris. (2009, October 24). CSS sprites: What they are, why they're cool, and how to use them. Retrieved July 1, 2013, from <http://css-tricks.com/css-sprites/>
- Coyle, K. (2007). The library catalog in a 2.0 world. *The Journal of Academic Librarianship*, 33(2), 289–291.
- Crawford, Susan P. (2013). *Captive audience: The telecom industry and monopoly power in the new gilded age*. New Haven, CT: Yale University Press.
- Croft, W. Bruce, Metzler, Donald, & Strohman, Trevor. (2010). Search engines: Information retrieval in practice. Boston, MA: Addison-Wesley.
- Csaplar, D. (2013). Optimizing business performance: Why every C-level manager should care about virtualization. Retrieved November 11, 2013 from <http://research.aberdeen.com/1/ebooks/virtualization/download/optimizing-business-performance.pdf>.
- css3html5. (2012, November 24). List of tablet and smartphone resolutions and screen sizes. Retrieved May 12, 2014, from <http://css3html5help.com/list-of-tablet-and-smartphone-resolutions-and-screen-sizes/>
- Cutts, Matt. (2013). About search. Retrieved July 22, 2013, from <http://www.google.com/competition/howgooglesearchworks.html>.
- Cyberthreat forecast for 2012. (2011). Retrieved June 13, 2013, from <http://www.kaspersky.com/images/Kaspersky%20report-10-134377.pdf>.
- Dankowski, Terra. (2013). How libraries are using social media. *American Libraries*, 44(5), 38–41.
- Data security with JSON. (2013, January 23). Retrieved December 12, 2013, from <https://www.golemtechnologies.com/articles/data-security-and-json>.
- Dave, Paresh. (2013, June 22). Wi-Fi is about to get a lot faster and more reliable. Retrieved July 18, 2013, from <http://articles.latimes.com/2013/jun/22/business/la-fi-tech-savvy-wifi-20130622>.
- Davis, Michele E., & Phillips, Jon A. (2007). *Learning PHP and MySQL* (2nd ed.). Sebastopol, CA: O'Reilly.
- de Kunder, M. (September 2008). The size of the World Wide Web. Retrieved September 21, 2008, from <http://www.worldwidewebsize.com>.
- Definition & usage—Wireframe. (2009, January 7). Retrieved June 21, 2013, from <http://web2usability.wordpress.com/2009/01/07/definition-usage-wireframe/>
- Deresiewicz, William. (2009a, January 30). The end of solitude. Retrieved September 5, 2013, from <http://chronicle.com/article/The-End-of-Solitude/3708>.

- Deresiewicz, William. (2009b, December 6). Faux friendship. Retrieved September 29, 2010, from <http://chronicle.com/article/Faux-Friendship/49308/>
- Desilver, Drew. (2013, May 20). 5 Facts about Tumblr. Retrieved September 4, 2013, from <http://www.pewresearch.org/fact-tank/2013/05/20/5-facts-about-tumblr/>
- Devine, Jane, & Egger-Sider, Francine. (2009). *Going beyond Google: The invisible Web in learning and teaching*. New York: Neal-Schuman Publishers, Inc.
- Dialog® Database Selection Guide. (2007). Retrieved September 18, 2013, from http://support.dialog.com/techdocs/co018002mi_dlg_dbselguide.pdf.
- Dibbell, J. (February, 2008). Griefer madness. *Wired*, 16, 90–97.
- Difference between MOV and MP4. (2013). Retrieved July 17, 2013, from <http://www.differencebetween.net/technology/difference-between-mov-and-mp4/>
- Difference between MOV vs AVI. (2013). Retrieved July 17, 2013, from <http://www.differencebetween.net/technology/difference-between-mov-vs-avi/>
- Dilger, Daniel Eran. (2013, May 14). Mobile malware exploding, but only for Android. Retrieved June 17, 2013, from <http://appleinsider.com/articles/13/05/14/mobile-malware-exploding-but-only-for-android>.
- Dillon, M., Jul, E., Burge, M., & Hickney, C. (1993). *Assessing information on the Internet: Toward providing library services for computer-mediated communication*. Dublin, OH: Online Computer Library Center, Inc.
- Ding, C. H., Nutanong, S., & Buyya, R. (2003). *Peer-to-peer networks for content sharing*. Melbourne, Australia: The University of Melbourne.
- Discover and learn. (2013). Retrieved May 30, 2013, from <http://www.wi-fi.org/discover-and-learn>.
- Doctor, Ken. (2012, April 12). The newconomics of small things. Retrieved April 16, 2013, from <http://www.niemanlab.org/2012/04/the-newconomics-of-small-things/>
- Dogpile. (April 2007). Different engines, different results, Retrieved June 1, 2008, from <http://www.infospaceinc.com/onlineprod/Overlap-DifferentEngines-DifferentResults.pdf>.
- Dragani, R. (2013, December 6). Stolen password analysis exposes foolish choices. Retrieved December 12, 2013, from <http://www.technewsworld.com/story/79574.html>.
- Dragland, Åse. (2013). Big data—for better or worse. Retrieved September 29, 2013 from <http://www.sintef.no/home/Press-Room/Research-News/Big-Data—for-better-or-worse/>
- Duggan, M., & Brenner, J. (2013, February 14). The demographics of social media users—2012. Retrieved May 19, 2014, from <http://www.pewinternet.org/2013/02/14/the-demographics-of-social-media-users-2012/>
- eBook formats. (2014). Retrieved April 18, 2014, from <http://ebookarchitects.com/learn-about-ebooks/format/>
- Eddy, W. M. (2006, December). Defenses against TCP SYN flooding attacks. Retrieved September 18, 2013, from http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_9-4/syn_flooding_attacks.html.
- EDUCAUSE. (2008). 7 Things you should know about Flickr. Retrieved May 30, 2008, from <http://net.educause.edu/ir/library/pdf/ELI7034.pdf>.
- Efros, A. (2003). Data-driven texture and motion. In *University of Washington Computer Science and Engineering colloquia*. [Video file]. Retrieved September 28, 2008, from <http://www.researchchannel.org/prog/displayevent.aspx?rID=3244&fID=345>.
- Electronic Privacy Information Center (EPIC). (2008, May 21). Congressman Barton urges scrutiny of Google's privacy practices. Retrieved June 1, 2008, from <http://epic.org>.

462 Bibliography

- Engard, Nicole C. (2009). *Library mashups: Exploring new ways to deliver library data*. Medford, NJ: Information Today, Inc.
- Enis, M. (2014). Putting the Pieces Together. *Library Journal*, 139(6), 32.
- Even, Loras R. (2000, July 12). Honey pot systems explained. Retrieved June 17, 2013, from <http://www.sans.org/security-resources/idfaq/honeypot3.php>.
- Expressing IPv6 addresses. (2005, January 21). Retrieved June 10, 2013, from <http://technet.microsoft.com/en-us/library/cc784831%28v=ws.10%29.aspx>.
- EZproxy® authentication and access software. (2014). Retrieved May 9, 2014, from <http://www.oclc.org/ezproxy.en.html>.
- Fahey, M. J., & Brown, J. W. (1995). *Web publisher's design guide for Windows*. Scottsdale, AZ: Coriolis Group.
- Fall, Kevin R., & Stevens, W. Richard. (2012). *TCP/IP illustrated* (2nd ed.). Upper Saddle River, NJ: Addison-Wesley.
- Farkas, M. (2007). Balancing the online life. *American Libraries*, 38(1), 42–45.
- Farrell, N. (2014, April 3). ICANN boss defends US giving up Internet ownership. Retrieved May 8, 2014, from <http://www.fudzilla.com/home/item/34393-icann-boss-defends-us-giving-up-internet-ownership>.
- Fay, R. M., & Sauers, M. (2012). *Semantic Web Technologies and Social Searching for Librarians*. Chicago, IL: ALA TechSource.
- Fichter, D. (2006). Using wikis to support online collaboration in libraries. *Information Outlook*, 10(1), 30–33.
- Fiocco, Alain. (2013, April). IPv6 adoption by the numbers. Retrieved June 7, 2013, from http://www.slideshare.net/getyourbuildon/naipv6-summi-2013t-ip_v6-adoption-by-the-numbers.
- Fisher, Tim. (2013). What is an FLV file? Retrieved July 16, 2013, from <http://pcsupport.about.com/od/fileextensions/f/flv-file.htm>.
- 56 K Info. (2007). Technical support. Retrieved May 1, 2008, from <http://home.core.com/web/technicalsupport/library/56kinfo.html>.
- Fling, Brian. (2009). *Mobile design and development* (1st ed.). Sebastopol, CA: O'Reilly.
- Flynn, R. R. (1987). *An introduction to information science*. New York: Marcel Dekker.
- FOAF vocabulary specification 0.98. (2010, August 9). Retrieved January 4, 2014, from <http://xmlns.com/foaf/spec/>
- 4G LTE. (2013). Retrieved June 4, 2013, from http://www.webopedia.com/TERM/4/4G_LTE.html.
- Frain, Ben. (2012). *Responsive Web design with HTML5 and CSS*. Birmingham, UK: Packt Publishing.
- Frazer, M. (2013, November 11). Building a dynamic image display with Drupal & Isotope. Retrieved November 22, 2013, from <http://acrl.ala.org/techconnect/?p=3739>.
- Friedman, T. L. (2005). *The world is flat: A brief history of the twenty-first century*. New York: Farrar, Straus, and Giroux.
- Gardner, L. D., & Grigsby, J. (2012). *Head first mobile web* (1st ed.). Sebastopol, CA: O'Reilly.
- Garfield, E. (1972). Citation analysis as a tool in journal evaluation. *Science*, 178(4060), 471–479.
- Gartner says 80 percent of active Internet users will have a “Second Life” in the virtual world by the end of 2011. (2007, April 24). Retrieved August 28, 2013, from <http://www.gartner.com/newsroom/id/503861>.
- Gartner says worldwide PC, tablet and mobile phone combined shipments to reach 2.4 billion units in 2013. (2013, April 4). Retrieved July 7, 2013, from <http://www.gartner.com/newsroom/id/2408515>.

- Gartner's 2009 hype cycle special report evaluates maturity of 1,650 technologies. (2009, August 11). Retrieved August 28, 2013, from <http://www.gartner.com/newsroom/id/1124212>.
- Gates, Bill. (1995). *The road ahead*. New York: Viking.
- Georgas, H. (2013). Google vs. the library: Student preferences and perceptions when doing research using Google and a federated search tool. *Libraries and the Academy*, 13(2), 165–185. Retrieved January 7, 2014, from Project MUSE database.
- Gesenhues, A. (2013, September 30). Google's hummingbird takes flight: SEOs give insight on Google's new algorithm. Retrieved December 30, 2013, from <http://searchengineland.com/hummingbird-has-the-industry-flapping-its-wings-in-excitement-reactions-from-seo-experts-on-googles-new-algorithm-173030>.
- Ghemawat, S., Gobioff, H., & Leung, S.-T. (2003, December). *The Google file system*. Paper presented at the Proceedings of the 19th ACM Symposium on Operating Systems Principles, Bolton Landing, New York.
- Gil, P. (August 2013). What is "the invisible Web"? Retrieved September 29, 2013, from <http://netforbeginners.about.com/cs/secondaryweb1/a/secondaryweb.htm>.
- Giles, J. (2005). Internet encyclopedias go head to head. *Nature*, 438, 900–901.
- Gill, J. (2008, January 17). Researchers' web use could make libraries redundant. Retrieved May 1, 2008, from <http://www.timeshighereducation.co.uk/story.asp?storycode=400168>.
- Gilroy, A.A., & Kruger, L.G. (2006). Broadband Internet regulation and access: Background and issues. Retrieved October 1, 2007, from <http://usinfo.state.gov/infousa/economy/technology/docs/60574.pdf>.
- Goldman, David. (2011, February 9). Smart phones have conquered PCs. Retrieved February 10, 2011, from http://money.cnn.com/2011/02/09/technology/smartphones_eclipse_pcs/index.htm?source=cnn_bin&hpt=Sbin.
- Goodrum, A., & Spink, A. (2001). Image searching on the Excite Web search engine. *Information Processing and Management*, 37, 295–311.
- Goodwin, Danny. (2013, January 15). Google Search market share slips as Bing & Yahoo gain. Retrieved July 22, 2013, from <http://searchenginewatch.com/article/2236637/Google-Search-Market-Share-Slips-as-Bing-Yahoo-Gain>.
- Google algorithm change history. (2013). Retrieved December 12, 2013, from <http://moz.com/google-algorithm-change>.
- Google. (2008). Mission statement. Retrieved May 1, 2008, from <http://www.google.com/corporate>.
- Graham, I. S. (1996). *The HTML sourcebook* (2nd ed.). New York: John Wiley & Sons.
- Graves, M. (2002). *Designing XML databases*. Upper Saddle River, NJ: Prentice Hall.
- Graziano, Dan. (2013, April 8). Google Fiber has cost less than \$100 million to launch so far from <http://bgr.com/2013/04/08/google-fiber-cost-nationwide-423644/>
- Griffee, Jason. (2010). *Mobile technology and libraries*. New York: Neal-Schuman Publishers.
- Griffith, Eric. (2011, November 29). Password protection: How to create strong passwords. Retrieved June 17, 2013, from <http://www.pcmag.com/article2/0,2817,2368484,00.asp>.
- Grimes, R. A. (2001). *Malicious mobile code*. Cambridge, MA: O'Reilly and Associates.
- Guide to implementing digital asset management systems and strategies. (2013). Retrieved November 24, 2013, from <http://searchcontentmanagement.techtarget.com/essentialguide/Guide-to-implementing-digital-asset-management-systems-and-strategies>.
- Gulli, A., & Signorini, A. (2005). *The indexable Web is more than 11.5 billion pages*. Paper presented at the WWW 2005, Chiba, Japan.

464 Bibliography

- Hafner, K. (2007, August 19). Seeing corporate fingerprints in Wikipedia edits. *New York Times*, p. 1.
- Hampton, Keith N., Goulet, Lauren Sessions, Rainie, Lee, & Purcell, Kristen. (2011, June 16). Social networking sites and our lives. Retrieved May 20, 2013, from <http://www.pewinternet.org/Reports/2011/~/media/Files/Reports/2011/PIP%20-%20Social%20networking%20sites%20and%20our%20lives.pdf>.
- Harms, D. (2001). *JSP, servlets, and MySQL*. New York: M&T Books.
- Harold, E. R., & Means, W. S. (2002). *XML in a nutshell* (3rd ed.). Beijing: O'Reilly.
- Harris, R. (2007, June 15). Evaluating Internet research sources. Retrieved October 1, 2007, from <http://www.virtualsalt.com/evalu8it.htm>.
- Hawking, D., & Zobel, J. (2007). Does topic metadata help with Web search? *JA-SIST*, 58(5), 613–626.
- The Heartbleed bug. (2014, April). Retrieved May 10, 2014, from <http://heartbleed.com/>
- Heise Security (2008, January 15). Quantity of malware booms. Retrieved May 1, 2008, from <http://www.heise-online.co.uk/security/Quantity-of-malware-booms—/news/101764>.
- Hempel, Jessi, Mansour, Iris, & Southward, Brandon. (2013, September 16). Social media all-stars. *Fortune*, 168, 133–139.
- Higginbotham, Stacey. (2009, April 17). Carriers aim to keep rural broadband under their thumb. Retrieved June 4, 2013, from <http://gigaom.com/2009/04/17/carriers-aim-to-keep-rural-broadband-under-their-thumb/>
- Higginbotham, Stacey. (2011, July 21). Since DSL is obsolete, AT&T will sell you LTE instead. Retrieved June 4, 2013, from <http://gigaom.com/2011/07/21/since-dsl-is-obsolete-att-will-sell-you-lte-instead/>
- History of PHP and related projects. (2014). Retrieved January 23, 2014, from <http://us2.php.net/history>.
- Hock, Randolph. (2013). *The extreme searcher's Internet handbook: A guide for the serious searcher* (4th ed.). Medford, NJ: CyberAge Books.
- Hoffer, J. A., Prescott, M. B., & McFadden, F. R. (2002). *Modern database management* (6th ed.). Upper Saddle River, NJ: Prentice Hall.
- Horaczek, Stan. (2013, May 27). How many photos are uploaded to the Internet every minute? Retrieved September 11, 2013, from <http://www.popphoto.com/news/2013/05/how-many-photos-are-uploaded-to-internet-every-minute>.
- Hughes, J. (2013, April 29). Micro service architecture. Retrieved November 24, 2013, from <http://yobriefca.se/blog/2013/04/29/micro-service-architecture/>
- Iasevoli, B. (2013, December 18). After bungled iPad rollout, lessons from LA put tablet technology in a time out. Retrieved January 23, 2014, from http://hechingerreport.org/content/after-bungled-ipad-rollout-lessons-from-la-put-tablet-technology-in-a-time-out_14123/
- Ingram, Mathew. (2012, April 16). The future of media=many small pieces, loosely joined. Retrieved April 16, 2013, from <http://www.businessweek.com/articles/2012-04-16/the-future-of-media>equals-many-small-pieces-loosely-joined>.
- Intel® Xeon Phi™ coprocessor 5110P: Highly parallel processing to power your breakthrough innovations (n.d.). Retrieved April 11, 2013, from <http://www.intel.com/content/www/us/en/processors/xeon/xeon-phi-detail.html>.
- Internet security report (2013, April). Retrieved January 31, 2014, from http://www.symantec.com/content/en/us/enterprise/other_resources/b-istr_main_report_v18_2012_21291018.en-us.pdf.
- Internet world users by language: Top 10 languages. (2011, May 31). Retrieved July 10, 2013, from <http://www.internetworldstats.com/stats7.htm>.

- Jacobs, B. (2010, September 24). How Shazam works to identify (nearly) every song you throw at it. Retrieved July 19, 2013, from <http://gizmodo.com/5647458/how-shazam-works-to-identify-nearly-every-song-you-throw-at-it>.
- Jeanneney, J.-N. (2006). *Google and the myth of universal knowledge*. Trans. T. L. Fagan. Chicago, IL: University of Chicago Press.
- Jones, Brandon. (2010, December 8). The Web designer's guide to comparing photoshop and fireworks. Retrieved July 1, 2013, from <http://webdesign.tutsplus.com/articles/general/the-web-designers-guide-to-comparingphotoshop-and-fireworks/>
- Jones, Kyle M. L., & Farrington, Polly-Alida. (2013). *Learning from libraries that use WordPress: Content-management system best practices and case studies*. Chicago, IL: American Library Association.
- Jones, S., & Madden, M. (2002, September 15). The Internet goes to college. *Internet and American Life Project*. Retrieved October, 2007, from http://www.pewinternet.org/pdfs/Pip_College_Report.pdf.
- Jowitt, A. L. (2008). Creating communities with podcasting. *Computers in Libraries*, 28(4), 14–15, 54–16.
- JSON Tutorial. (2013). Retrieved December 9, 2013, from <http://www.w3schools.com/json/default.asp>.
- Jupiter Media. (2005). Jupiter research forecasts broadband's rise to dominance, increasing from 32 million U.S. households in 2004 to 69 million in 2010. Retrieved October, 2007, from <http://www.jupitermedia.com/corporate/releases/05.06.02-newjupresearch.html>.
- Kadlec, Tim. (2013). *Implementing responsive design: Building sites for an anywhere, everywhere web*. Berkeley, CA: New Riders.
- Kahle, B. (1997). Preserving the Internet. *Scientific American*, 276(3), 82–84.
- Kang, Cecilia. (2012, January 24). Google announces privacy changes across products; users can't opt out. Retrieved July 19, 2013, from http://www.washingtonpost.com/business/economy/google-tracks-consumers-across-products-users-cant-opt-out/2012/01/24/gIQArgJHOQ_story.html.
- Kaplan, Andreas M., & Haenlein, Michael. (2010). Users of the world, unite! The challenges and opportunities of social media. *Business Horizons*, 53(1), 59–68. doi: <http://dx.doi.org/10.1016/j.bushor.2009.09.003>.
- Kaplenk, J. (1999). *UNIX system administrator's interactive workbook*. Upper Saddle River, NJ: Prentice-Hall.
- Kelley, K., Leatherman, C. C., & Rinna, G. (2013). Is it really time to replace your ILS with a next-generation option? *Computers in Libraries*, 33(8), 11–15.
- Kerner, Sean Michael. (2013, September 13). Apple updates Mac OS X 10.8.5 for security, stability. Retrieved September 28, 2013, from <http://www.eweek.com/security/apple-updates-mac-os-x-10.8.5-for-security-stability.html>.
- Kestenbaum, D. (2008). Old drug offers new hope for Marfan syndrome. Retrieved May 30, 2008, from <http://www.npr.org/templates/story/story.php?storyId=90257827>.
- Khalaf, S. (2014, January 13). Mobile use grows 115% in 2013, propelled by messaging apps. Retrieved January 15, 2014, from <http://blog.flurry.com/>
- Kiernan, V. (2005). Missing the boat, or penny-wise caution? *The Chronicle of Higher Education*, 51(27), A33–35.
- King, R. (2012, July 5). Netflix streaming tops 1 billion hours in month for first time. Retrieved September 8, 2013, from http://news.cnet.com/8301-1023_3-57467191-93/netflix-streaming-tops-1-billion-hours-in-month-for-first-time/
- Klein, N. (2008, May 29). China's all-seeing eye: With the help of U.S. defense contractors, China is building the prototype for a high-tech police state. Retrieved

466 Bibliography

- May 30, 2008 from http://www.rollingstone.com/politics/story/20797485/chinas_allseeing_eye.
- Kleinberg, J. M. (1999). Authoritative sources in a hypertext environment. *Journal of the ACM*, 46(5), 604–632.
- Kleinrock, L. (1996, August 27). The birth of the Internet. Retrieved October 1, 2007, from <http://www.lk.cs.ucla.edu/LK/Inet/birth.html>.
- Klosowsk, Thorin (2013, January 7). How Web sites vary prices based on your information (and what you can do about it). Retrieved September 26, 2013, from <http://lifehacker.com/5973689/how-web-sites-vary-prices-based-on-your-information-and-what-you-can-do-about-it>.
- Kolowich, S. (2013, May 21). MOOC professors claim no responsibility for how courses are used. *The Chronicle of Higher Education*. Retrieved June 5, 2013, from <http://chronicle.com/blogs/wiredcampus/mooc-professors-claim-no-responsibility-for-how-courses-are-used/43881>.
- Korfhage, R. R. (1997). *Information storage and retrieval*. New York: Wiley & Sons.
- Kozierok, Charles M. (2005, September 30). The TCP/IP guide. Retrieved June 11, 2013, from http://www.tcpipguide.com/free/t_IPv6DatagramOverviewandGeneralStructure.htm.
- Krishnamurthy, Balachander, Mogul, Jeffrey C., & Kristol, David M. (1999). Key differences between HTTP/1.0 and HTTP/1.1. Retrieved June 11, 2013, from <http://www8.org/w8-papers/5c-protocols/key/key.html>.
- Krol, E. (1994). *The whole Internet: Users guide and catalog*. Sebastopol, CA: O'Reilly and Associates.
- Kroski, E. (2007, April 2). Information design for the new Web. Retrieved October 1, 2007, from <http://infotangle.blogspot.com/2007/04/02/information-design-for-the-new-web>.
- Krug, Steve. (2006). *Don't make me think!: A common sense approach to Web usability* (2nd ed.). Berkeley, CA: New Riders Pub.
- Kyrnin, Jennifer. (2006, November 22). Use the Google Transcoder to simplify your pages. Retrieved June 19, 2013, from <http://webdesign.about.com/b/2006/11/22/use-the-google-transcoder-to-simplify-your-pages.htm>.
- KZERO Worldwide. (2012, January 6). Virtual worlds: Industry and user data. Retrieved September 29, 2013, from http://www.slideshare.net/nicmitham/kzero-universe-q4-2011?from=ss_embed.
- La Counte, Scott. (2012). *Going mobile: Developing apps for your library using basic HTML programming*. Chicago, IL: ALA Editions.
- Langville, A. N. (2005). The linear algebra behind search engines. Retrieved October 15, 2007, from <http://mathdl.maa.org/mathDL/4/?pa=content&sa=viewDocument&nodeId=636>.
- Langville, A. N., & Meyer, C. D. (2006). *Google's PageRank and beyond: The science of search engine rankings*. Princeton, NJ: Princeton University Press.
- Lankes, R. David. (2011). *The atlas of new librarianship*. Cambridge, MA: MIT Press.
- Larson, E. (2013, May 30). 5 Reasons to choose Vimeo instead of YouTube. Retrieved November 25, 2013, from <http://mashable.com/2013/05/30/vimeo-over-youtube/>
- Lawrence, S., & Giles, C. L. (1998). Searching the World Wide Web. *Science*, 280(5360), 98–100.
- Lawrence, S., & Giles, C. L. (1999). Accessibility of information on the Web. *Nature*, 400(6740), 107–109.
- Lawrence, S., Coetzee, F., Glover, E., Flake, G., Pennock, D., Krovetz, B., et al. (2000). *Persistence of information on the web: Analyzing citations contained in research articles*. Paper presented at the Conference on Information and Knowledge Management, McLean, Virginia.

- Lawson, Bruce, & Sharp, Remy. (2011). *Introducing HTML5*. Berkeley, CA: New Riders.
- Layout, Kristofer. (2012). *Mobilizing web sites: Develop and design*. Berkeley, CA: Peachpit.
- Learn about Java technology. (2013). Retrieved May 23, 2013, from <http://www.java.com/en/about/>
- Leiner, B.M., Cerf, V.G., Clark, D.D., Kahn, R.E., Kleinrock, L., Lynch, D.C., et al. (1997). The past and future history of the Internet. *Communications of the ACM*, 40(2), 102–109.
- LeMay, L. (1997). *Teach yourself Web publishing with HTML 4* (2nd ed.). Indianapolis, IN: SAMS.
- Lengstorf, J. (2009). JSON: What it is, how it works, & how to use it. Retrieved December 9, 2013, from <http://www.copterlabs.com/blog/json-what-it-is-how-it-works-how-to-use-it/>
- Lenhart, A., & Fox, S. (2006, July 19). Bloggers: A portrait of the Internet's new storytellers. Retrieved October 1, 2007, from <http://www.pewinternet.org/pdfs/PIP%20Bloggers%20Report%20July%2019%202006.pdf>.
- LibGuides FAQ. (2013). Retrieved July 7, 2013, from <http://www.springshare.com/libguides/benefits.html>.
- Lie, H.W., & Bos, B. (1999). *Cascading style sheets, designing for the Web*. Reading, MA: Addison Wesley.
- Loshin, P. (1999). *IPv6 clearly explained*. San Francisco, CA: Academic Press.
- Lumsden, Aaron. (2012, June 27). Getting started with scalable vector graphics (SVG). Retrieved June 21, 2013, from <http://webdesign.tutsplus.com/tutorials/htmlcss-tutorials/getting-started-with-scalable-vector-graphics-svg/>
- Lynn, Samara. (2010, September 7). Ten tips for public Wi-Fi hotspot security. Retrieved June 17, 2013, from <http://www.pc当地.com/article2/0,2817,2368802,00.asp>.
- Lynn, Samara. (2012, November 8). How to buy a wireless router. Retrieved May 30, 2013, from <http://www.pc当地.com/article2/0,2817,2347539,00.asp>.
- MacVittie, Lori. (2013). *ScaleN: Elastic infrastructure*. Seattle, WA: F5 Networks.
- Madden, Mary. (2010, August 27). Older adults and social media. Retrieved September 3, 2013, from <http://www.pewinternet.org/Reports/2010/Older-Adults-and-Social-Media.aspx>.
- Madden, Mary, Lenhart, Amanda, Cortesi, Sandra, Gasser, Urs, Duggan, Maeve, Smith, Aaron, & Beaton, Meredith (2013, May 21). Teens, social media, and privacy. from <http://www.pewinternet.org/Reports/2013/Teens-Social-Media-And-Privacy/Main-Report/Part-1.aspx>.
- Malik, Om. (2012, January 6). The continued decline of DSL. Retrieved June 3, 2013, from <http://gigaom.com/2012/01/26/the-continued-decline-of-dsl/>
- Mann, T. (2006, April 3). The changing nature of the catalog and its integration with other discovery tools: A critical review. Retrieved November 18, 2013, from <http://www.guild2910.org/AFSCMCECalhounReviewREV.pdf>.
- Marcotte, Ethan. (2010). Responsive Web design. Retrieved May 4, 2012, from <http://alistapart.com/article/responsive-web-design>.
- Marcotte, Ethan. (2011). *Responsive Web design*. New York: A Book Apart.
- Marquis, Mat. (2012, January 31). Responsive images: How they almost worked and what we need. Retrieved June 21, 2013, from <http://alistapart.com/article/responsive-images-how-they-almost-worked-and-what-we-need>.
- Martin, D., Prata, S., Waite, M., Wessler, M., & Wilson, D. (2000). *UNIX primer plus* (3rd ed.). Indianapolis, IN: Sams.
- Mathews, Brian. (2010, May 19). Redesigning your website? Why not use LibGuides as your content management system? Retrieved July 12, 2013, from <http://>

- theubiquitouslibrarian.typepad.com/the_ubiquitous_librarian/2010/05/re-designing-your-website-why-not-use-libguides-as-your-content-management-system.html.
- McCall, T. (2007, December 17). Gartner survey shows phishing attacks escalated in 2007; more than \$3 billion lost to these attacks. Retrieved June 18, 2013, from <http://www.gartner.com/it/page.jsp?id=565125>.
- McClelland, D., Eismann, K., & Stone, T. (2000). *Web design studio secrets* (2nd ed.). Foster City, CA: IDG Books.
- McFarland, David Sawyer. (2011). *JavaScript & jQuery* (2nd ed.). Sebastopol, CA: O'Reilly.
- McGee, Matt. (2013, May 15). Bing rises above 17% search market share as Google slips [comScore]. Retrieved July 19, 2013, from <http://searchengineland.com/bing-rises-above-17-search-market-share-as-google-slips-com-score-159746>.
- McIntyre, Douglas A. (2009, March 9). The ten major newspapers that will fold or go digital next. Retrieved April 16, 2013, from <http://247wallst.com/2009/03/09/the-ten-major-newspapers-that-will-fold-or-go-digital-next/>
- Meadow, C. T., Boyce, B. R., & Kraft, D. H. (2000). *Text information retrieval systems* (2nd ed.). San Diego, CA: Academic Press.
- Meadows, Chris. (2013). A look at the FLV file. Retrieved July 16, 2013, from <http://pc.answers.com/computer-science/a-look-at-the-flv-file>.
- Mearian, Lucas. (2009, January 2). Back to the future: Vinyl record sales double in '08, CDs down. Retrieved April 16, 2013, from http://www.computerworld.com/s/article/9124699/Back_to_the_future_Vinyl_record_sales_double_in_08_CDs_down.
- Meloni, J. C. (2000). *PHP fast & easy Web development*. Roseville, CA: Prima Publishing.
- MessageLabs March 2011 intelligence report. (2011, March). Retrieved June 11, 2013, from http://www.message-labs.com/mlireport/MLI_2011_03_March_Final-EN.pdf.
- Mi, J., & Weng, C. (2008). Revitalizing the library OPAC: Interface, searching and display challenges. *Information Technology and Libraries*, 27(1), 5–22.
- Michel, Jason Paul. (2012). *Web service APIs and libraries*. Chicago, IL: American Library Association.
- Microsoft.NET. (2014). Overview. Retrieved January 23, 2014 from <http://msdn.microsoft.com/en-us/library/zw4w595w.aspx>.
- Miller, Rich. (2011, August 1). Google uses about 900,000 servers. Retrieved September 22, 2013, from <http://www.datacenterknowledge.com/archives/2011/08/01/report-google-uses-about-900000-servers/>
- Miller, S. J. (2011). *Metadata for digital collections: A how-to-do-it manual*. New York: Neal-Schuman Publishers.
- Milliot, Jim. (2012, November 5). E-books market share at 22%, Amazon has 27%. Retrieved September 12, 2013, from <http://www.publishersweekly.com/pw/by-topic/digital/retailing/article/54609-e-books-market-share-at-22-amazon-has-27.html>.
- Mills, Chris. (2013, April 23). Five simple ways to keep your Android malware-free. Retrieved June 17, 2013, from <http://gizmodo.com/5995254/five-simple-ways-to-keep-your-android-malware-free>.
- Mills, E. (2005, October 21). Google shares soar on hearty revenue report. Retrieved October 1, 2007, from http://news.com.com/Google+revenue+nearly+doubles/2100-1030_3-5905127.html.
- Mitchell, Bradley. (2013a). SNMP. Retrieved June 12, 2013, from <http://compnetworking.about.com/od/networkprotocols/g/snmp-management-protocol.htm>.

- Mitchell, Bradley. (2013b). LDAP. Retrieved June 12, 2013, from <http://compnetworking.about.com/library/glossary/bldef-ldap.htm>.
- Mitchell, J. (2014). Beyond the Maker Space. *Library Journal*, 139(9), 37.
- Mobile majority: U.S. smartphone ownership tops 60%. (2013, June 6). Retrieved September 13, 2013, from <http://www.nielsen.com/us/en/newswire/2013/mobile-majority—u-s—smartphone-ownership-tops-60-.html>.
- Mollenkopf, Jim. (2004, September 23). Presentation to Cincinnati IEEE meeting. Retrieved July 18, 2013 from http://ieee.cincinnati.fuse.net/BPL_slide_show.pdf.
- Molyneux, R. E. (2003). *The Internet under the hood: An introduction to network technologies for information professionals*. Westport, CT: Libraries Unlimited.
- Moorman, Christine. (2013, September 3). Big data's big puzzle. Retrieved September 10, 2013, from <http://www.forbes.com/sites/christinemoorman/2013/09/03/big-datas-big-puzzle/>
- More than one third of web pages are pornographic. (2010, June 16). Press release. Retrieved July 18, 2013, from <http://www.optenet.com/en-us/news.asp?id=270>.
- Morgantini, D. (2013, August 27). Micro-services. Retrieved December 2, 2013, from <http://davidmorgantini.blogspot.com/2013/08/micro-services-why-shouldnt-you-use.html>.
- Morris, Chris. (2012, October 24). The 25 most popular passwords of 2012. Retrieved June 18, 2013, from <http://games.yahoo.com/blogs/plugged-in/25-most-popular-passwords-2012-164015152.html>.
- Morville, P. (2005). *Ambient findability*. Sepastpol, CA: O'Reilly Media, Inc.
- Mowshowitz, A., & Kawaguchi, A. (2002). Bias on the Web. *Communications of the ACM*, 45(9), 56–60.
- Murthy, Dhiraj. (2013). *Twitter: Social communication in the Twitter age*. Cambridge: Polity.
- National Science Foundation. (2005, September 28). The launch of NSFNET. Retrieved October 1, 2007, from <http://www.nsf.gov/about/history/nsf0050/internet/launch.htm>.
- Needham, M. (2012, November 29). Micro services: The curse of code duplication. Retrieved November 25, 2013, from <http://architects.dzone.com/articles/micro-services-curse-code>.
- Nielsen Company. (2008, May 14). Wikipedia U.S. Web traffic grows 8,000 percent in five years, driven by search. Retrieved June 10, 2008, from http://www.nielsen-netratings.com/pr/pr_080514.pdf.
- Nielsen, J. (2003, June 30). Information foraging: Why Google makes people leave your site faster. Retrieved October 1, 2007, from <http://www.useit.com/alertbox/20030630.html>.
- Nielsen, J. (2006, April 17). F-shaped pattern for reading Web content. Retrieved July 9, 2013, from <http://www.nngroup.com/articles/f-shaped-pattern-reading-web-content/>
- Nielsen, J. (2008, April 28). Right-justified navigation menus impede scannability. Retrieved July 10, 2013 from <http://www.useit.com/alertbox/navigation-menu-alignment.html>.
- Norman, D. (2007). Simplicity is highly overrated. Retrieved October 1, 2007, from http://www.jnd.org/dn.mss/simplicity_is_highly.html.
- Notess, G. (2013). Search engine to knowledge engine? *Online Searcher*, 37(4), 3.
- Notess, G.R. (2002, March 6). Little overlap despite growth! Retrieved October 1, 2007, from <http://www.searchengineshowdown.com/statistics/overlap.shtml>.
- Online Computer Library Center (OCLC). (2002). Web characterization. Retrieved October 1, 2007, from <http://www.oclc.org/research/projects/archive/wcp>.

470 Bibliography

- Online Community Library Center (OCLC). (2006, June 1). OCLC reports on college students' library perceptions. Retrieved May 1, 2008, from <http://www.libraryjournal.com/info/CA6340281.html>.
- O'Reilly Media, Inc. (2008). Distributed search engines. Retrieved October 1, 2007, from <http://www.openp2p.com/pub/t/74>.
- O'Reilly, T. (2005, October 1). Web 2.0: Compact definition? Retrieved October 1, 2007, from <http://radar.oreilly.com/archives/2005/10/web-20-compact-definition.html>.
- O'Reilly, T. (2006, May 14). My commencement speech at SIMS. Retrieved June 2, 2008 from <http://radar.oreilly.com/archives/2006/05/my-commencement-speech-at-sims.html>.
- O'Rourke, Dave. (2012). Camtasia Studio support of MP4, SWF, and FLV file formats. Retrieved July 16, 2013, from http://feedback.techsmith.com/techsmith/topics/camtasia_studio_support_of_mp4_swf_and_flv_file_formats.
- Oswald, Ed. (2012, May 9). Demystifying semantic search. Retrieved July 19, 2013, from <http://www.extremetech.com/computing/123599-demystifying-semantic-search>.
- Otis, B., & Parviz, B. (2014, January 16). Introducing our smart contact lens project. Retrieved January 23, 2014, from <http://googleblog.blogspot.com/2014/01/introducing-our-smart-contact-lens.html>.
- Parsons, June, & Oja, Dan. (2012). *NP on Computer Concepts 2013 comprehensive*. Boston, MA: Cengage Course Technology.
- Pepus, G. (2007). Smart image and video search. *KM World*, 16(6), 6–9.
- Perez, Sarah. (2013, February 25). Apple rejecting apps using cookie-tracking methods, signaling push to its own ad identifier technology is now underway. Retrieved September 26, 2013, from <http://techcrunch.com/2013/02/25/apple-rejecting-apps-using-cookie-tracking-methods-signaling-push-to-its-own-ad-identifier-technology-is-now-underway/>
- Perry, Mark J. (2012, February 26). Newspaper ad revenues fall to 60-yr. low in 2011. Retrieved April 16, 2013, from <http://mjperry.blogspot.ca/2012/02/newspaper-ad-revenues-fall-to-50-year.html>.
- Peters, Dave. (2011, July 12). Broadband 7 update: Fiber hard to find and bids high but construction starts. Retrieved August 15, 2013, from <http://blogs.mprnews.org/ground-level/2011/07/broadband-7-revisited/>
- Peters, T., & Bell, L. (2008, April 11). Trends, fads or folly: Spotting the library trends that really matter. Retrieved April 15, 2008, from <http://www.collegeofdupagepress.com/library-learning-network/soaring-to-excellence-2008/spotting-the-library-trends-that-really-matter/>
- Petersen, Steve. (2012, February 15). 10Drupalfundamentals. Retrieved July 19, 2013, from <http://blog.thebrickfactory.com/2012/02/10-drupal-fundamentals/>
- Pfeifer, R., & Bongard, J. C. (2007). *How the body shapes the way we think*. Cambridge: MIT Press.
- Pogue, David. (2012, February 28). Use it better: 8 Alternatives to the hated Captcha. Retrieved June 19, 2013, from <http://www.scientificamerican.com/article.cfm?id=pogue-8-alternatives-to-hated-captcha>.
- Powell, T. A. (2003). *HTML and XHTML: The complete reference*. Emeryville, CA: McGraw-Hill/Osborne.
- Powers, D. (2006). *PHP Solutions: Dynamic Web design made easy*. New York: Springer-Verlag.
- Pressman, Aaron. (2014, January 16). More than 3 million teens flee Facebook. Retrieved January 19, 2014 from <http://finance.yahoo.com/blogs/the-exchange/about-11-million-fewer-likes-for-facebook-among-the-teenage-set-154852067.html>.

- Purcell, Kristen. (2011). Search and email still top the list of most popular online activities Retrieved June 11, 2013, from <http://www.pewinternet.org/2011/08/09/search-and-email-still-top-the-list-of-most-popular-online-activities/>
- Purcell, Kristen, Brenner, Joanna, & Rainie, Lee. (2012, March 9). Search engine use 2012. Retrieved September 11, 2013, from <http://www.pewinternet.org/Reports/2012/Search-Engine-Use-2012.aspx>.
- Purvis, M., Sambells, J., & Turner, C. (2006). *Beginning Google maps applications with PHP and Ajax: from novice to professional*. Berkeley, CA: Apress.
- Rainie, L. (2005, November 20). Search engine use shoots up in the past year and edges towards email as the primary internet application. *Internet and American Life*. Retrieved May 1, 2008, from http://www.pewinternet.org/PPF/r/167/report_display.asp.
- Rainie, L. (2007, January 31). Tagging play: Forget Dewey and his decimals, Internet users are revolutionizing the way we classify information—and make sense of it. *Pew Internet & American Life Project*. Retrieved May 30, 2008 from <http://pewresearch.org/pubs/402/tagging-play>.
- Raiu, Costin, & Emm, David. (2012, December 5). Kaspersky Security Bulletin 2012. Malware evolution. Retrieved June 13, 2013, from <http://www.securelist.com/en/analysis/204792254/>
- Rapacki, S. (2007). Why teens need places like MySpace. *Young Adult Library Services*, 5(2).
- RDFa. (2013, October 28). Retrieved January 4, 2014, from <http://wiki.creativecommons.org/RDFa>.
- RDFa 1.1 primer—Second edition: Rich structured data markup for Web documents. (2013, August 22). Retrieved January 4, 2014, from <http://www.w3.org/TR/xhtml-rdfa-primer/>
- Reed, B. (2012, September 28). How Google wants to make Google Now a psychic stalker version of Siri. Retrieved December 26, 2013, from <http://bgr.com/2012/09/28/google-now-features-preemptive-answers/>
- Reference model for an open archival information system (OAIS): Recommended practice. (2012, June). Retrieved November 25, 2013, from <http://public.ccsds.org/publications/archive/650x0m2.pdf>.
- Reiss, S. (2008, May, 2008). Planet Amazon. *Wired*, 16, 88–95.
- Resource description framework (RDF) model and syntax specification. (1999, January 5). Retrieved January 4, 2014, from <http://www.w3.org/TR/PR-rdf-syntax/>
- Rethlefsen, M. L. (2007). Tags help make libraries DELICIO.US. *Library Journal*, 131(15), 26–28.
- Riccardi, G. (2003). *Database management with Web site development applications*. New York: Addison Wesley.
- Riddell, R. (2013, February 6). 12 Learning management system providers and what they bring to classrooms. Retrieved January 20, 2014, from <http://www.educationdive.com/news/12-learning-management-system-providers-and-what-they-bring-to-classrooms/97613/>
- Rob, P., & Coronel, C. (1997). *Database systems: Design, implementation, and management* (3rd ed.). Cambridge, MA: Course Technology.
- Roberson, Jarrod. (2010, January 15). Is HTTP/1.0 still in use? Retrieved June 10, 2013, from <http://stackoverflow.com/questions/2073392/is-http-1-0-still-in-use>.
- Rogers, I. (2002). The Google Pagerank algorithm and how it works. *Google Pagerank whitepaper*. Retrieved October 10, 2007, from <http://www.ianrogers.net/google-page-rank>.

472 Bibliography

- Rorissa, A. (2007). Benchmarking visual information indexing and retrieval systems. *Bulletin of the American Society for Information Science and Technology*, 33(3), 15–17.
- Rosen, Rebecca J. (2012, June 27). 59% of Young people say the Internet is shaping who they are. Retrieved May 1, 2013, from <http://www.theatlantic.com/technology/archive/2012/06/59-of-young-people-say-the-internet-is-shaping-who-they-are/259022/>
- Rosenfeld, L., & Morville, P. (1998). *Information architecture for the World Wide Web*. Sebastopol, CA: O'Reilly.
- Rubenking, Neil. J. (2010, April 7). Analyst's View: PDF—Pretty Dangerous Format? Retrieved May 1, 2014, from <http://www.pcmag.com/article2/0,2817,2362356,00.asp>.
- Rubenking, Neil J. (2013, March 20). 7 Signs you've got malware. Retrieved June 13, 2013, from <http://www.pcmag.com/article2/0%2C2817%2C2416788%2C00.asp>.
- Ruby on Rails. (2013). Web development that doesn't hurt. Retrieved September 30, 2013, from <http://www.rubyonrails.org>.
- Ruotsalo, T. (2012). Domain specific data retrieval on the semantic Web. In E. Simperl, P. Cimiano, A. Polleres, O. Corcho, & V. Presutti (Eds.), *The Semantic Web: Research and applications* (vol. 7295, pp. 422–436). Berlin, Heidelberg: Springer.
- Russo, P., & Boor, S. (1993, April 24–29). *How fluent is your interface? Designing for international users*. Paper presented at the INTERCHI '93, Amsterdam, The Netherlands.
- Ruvolo, Julie. (2011, September 7). How much of the Internet is actually for porn. Retrieved July 18, 2013, from <http://www.forbes.com/sites/julieruvolo/2011/09/07/how-much-of-the-internet-is-actually-for-porn/>
- Salton, G., & McGill, M.J. (1983). *Introduction to modern information retrieval*. New York: McGraw-Hill.
- Satariano, Adam (2013, September 11). Tablet shipments to exceed personal computers. Retrieved September 12, 2013, from http://www.bloomberg.com/news/2013-09-11/tablet-shipments-to-exceed-personal-computers.html?source=email_rt_mc_body&app=n.
- Saul, D. (2014, January 15). 3 Million teens leave Facebook in 3 years: The 2014 Facebook demographic report. Retrieved January 19, 2014 from <http://istrategylabs.com/2014/01/3-million-teens-leave-facebook-in-3-years-the-2014-facebook-demographic-report/>
- Scheeren, William O. (2012). *The hidden Web: A sourcebook*. Santa Barbara, CA: Libraries Unlimited, an imprint of ABC-CLIO, LLC.
- Schneider, G. M., & Gersting, J. L. (2000). *An invitation to computer science*. Pacific Grove, CA: Brooks/Cole.
- Schrock, K. (1998). 5 W's for evaluating Web sites. Retrieved October 1, 2007, from <http://kathyschrock.net/abceval/5ws.htm>.
- Schull, Diantha Dow. (2013). *50+ Library services: Innovation in action*. Chicago, IL: ALA Editions.
- Schwartz, B. (2005). *The paradox of choice: Why more is less*. New York: Harper Perennial.
- Schwartz, H. Why CCM is not a CMS: Or why you shouldn't confuse a whale with a fish. Retrieved November 25, 2013 from <http://www.infomanagementcenter.com/members/pdfs/reprints/BP09-12HSchwartz.pdf>.
- Security TechCenter. (2014, May 1). Security update for Internet Explorer (2965111). Retrieved May 11, 2014, from <https://technet.microsoft.com/library/security/ms14-021>.

- SFX—The OpenURL link resolver and much more. (2012). Retrieved September 16, 2013, from <http://www.exlibrisgroup.com/category/SFXOverview>.
- Shafer, D. (1996). *JavaScript & Netscape wizardry*. Scottsdale, AZ: Coriolis Group Books.
- Shannon, C. (1949). Communication in the presence of noise. *Proceedings of the Institute of Radio Engineers*, 37(1), 10–21.
- Shannon, V. (2006, May 24). A “more revolutionary” Web. Retrieved May 1, 2008, from <http://www.iht.com/articles/2006/05/23/business/web.php>.
- Sheldon, T. (2001). DSL (Digital Subscriber Line). *Tom Sheldon’s Linktionary*. Retrieved January 27, 2006, from <http://www.linktionary.com/d/dsl.html>.
- Simoneau, P. (1997). *Hands-on TCP/IP*. New York: McGraw-Hill.
- Singhal, Amit, & Cutts, Matt. (2011, February 24). Finding more high-quality sites in search. Retrieved July 22, 2013, from <http://googleblog.blogspot.com/2011/02/finding-more-high-quality-sites-in.html>.
- Sivonen, Henri. (2013, March 9). Activating browser modes with doctype. Retrieved June 24, 2013, from <http://hsivonen.iki.fi/doctype/>
- Sloan, S. (2006, February 19). Inklings of change—Inside the potential sale of Knight Ridder: Industry pressures, technology advances put print journalism and Herald-Leader at a crossroads. *Lexington Herald Leader*, p. A16.
- Smalera, P. (2007, September). Google’s secret formula. *Conde Nast Portfolio*, 136–137.
- Smeaton, A. (2007). TRECVID-video evaluation. *Bulletin of the American Society for Information Science and Technology*, 33(3), 21–23.
- Smith, Aaron. (2010, July 7). Mobile access 2010. Retrieved May 30, 2013, from <http://www.pewinternet.org/Reports/2010/Mobile-Access-2010.aspx>.
- Smith, Aaron. (2012, March 1). Nearly half of American adults are smart phone owners. Retrieved May 30, 2013, from <http://pewinternet.org/Reports/2012/Smartphone-Update-2012.aspx>.
- Smith, Aaron. (2013, January 31). In-store mobile commerce during the 2012 holiday shopping season. Retrieved May 22, 2013, from <http://pewinternet.org/Reports/2013/in-store-mobile-commerce.aspx>.
- Smith, C. (2010, October 12). What’s best: Microformats, RDFa, or micro data? Retrieved January 27, 2014, from <http://www.semclubhouse.com/micro-formats-rdfa-or-micro-data/>
- SMS triples in three years. (2011). Retrieved September 19, 2013, from <http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf>.
- Soloman, Laura. (2013). *The librarian’s nitty-gritty guide to social media*. Chicago, IL: ALA Editions.
- Soltani, Ashkan, Peterson, Andrea, & Gellman, Barton. (2013, December 10). NSA uses Google cookies to pinpoint targets for hacking. Retrieved January 5, 2014, from <http://www.washingtonpost.com/blogs/the-switch/wp/2013/12/10/nsa-uses-google-cookies-to-pinpoint-targets-for-hacking/>
- Sophos, Inc. (2002, May 1). Melissa worm author sentenced to 20 months. Retrieved June 18, 2013, from http://www.sophos.com/pressoffice/news/articles/2002/05/pr_uk_20020501smith.html.
- Sophos, Inc. (2006, June 8). JDBGMGR hoax. Retrieved June 18, 2013, from <http://www.sophos.com/en-us/threat-center/threat-analyses/hoaxes/virus-hoax/jdbgmgr.aspx>.
- Sperberg-McQueen, C. M., & Thompson, H. (2008, July). W3C XML Schema. Retrieved September 27, 2008 from <http://www.w3.org/XML/Schema>.
- Sriparasa, S. S. (2013). *JavaScript and JSON essentials paperback*. Birmingham, UK: Packt Publishing.

474 Bibliography

- State of the media: The social media report 2012. (2012). Retrieved July 7, 2013, from <http://www.nielsen.com/us/en/reports/2012/state-of-the-media-the-social-media-report-2012.html>.
- The state of the news media 2013—An annual report on American journalism. (2013). Retrieved January 4, 2014, from <http://stateofthemedia.org/2013/overview-5/key-findings/economics-key-findings-7/#save-chart>.
- Stelter, B. (2008, June 15). Internet providers clamp down on users' time online. *Lexington Herald-Leader*, p. A7.
- Stenovec, Timothy. (2013, June 12). MySpace relaunch: Redesigned site, mobile app include MyRadio streaming music service. Retrieved August 14, 2013, from http://www.huffingtonpost.com/2013/06/12/myspace-relaunch-my-radio-app_n_3428934.html.
- Stephens, M. (2007). *Web 2.0 & libraries, part 2: Trends and technologies* (Report). Chicago, IL: American Library Association.
- Stoll, Clifford. (1995). *Silicon snake oil: Second thoughts on the information highway* (1st ed.). New York: Doubleday.
- Storey, T. (2005). The long tail and libraries. *OCLC Newsletter*, 268, 6–10.
- Strange, A. (2014, April 8). Widespread encryption bug, Heartbleed, can capture your passwords. Retrieved May 10, 2014, from <http://mashable.com/2014/04/08/major-security-encryption-bug-heartbleed/>
- Suetos, Shannon. (2010, April 26). HTML5: Worth the hype? Retrieved June 24, 2013, from <http://www.instantshift.com/2010/04/26/html5-worth-the-hype/>
- Sullivan, D. (2004, January 6). Google's (and Inktomi's) miserable failure. Retrieved May 1, 2008, from <http://searchenginewatch.com/showPage.html?page=3296101>.
- Sullivan, D. (2005, January 28). Search engine sizes. Retrieved October 1, 2007, from <http://searchenginewatch.com/showPage.html?page=2156481#trend>.
- Sullivan, D. (2011, February 1). Google: Bing is cheating, copying our search results. Retrieved December 14, 2013, from <http://searchengineland.com/google-bing-is-cheating-copying-our-search-results-62914>.
- Sullivan, D. (2013, October 4). Penguin 5, with the Penguin 2.1 spam-filtering algorithm, is now live. Retrieved December 12, 2013, from <http://searchengineland.com/penguin-2-1-and-5-live-173632>.
- SVG Working Group. (2003, January 14). Scalable vector graphics (SVG) 1.1 specification: W3C recommendation. Retrieved May 1, 2008, from <http://www.w3.org/TR/SVG>.
- Symantec Internet security threat report. (2011). Retrieved January 31, 2014, from http://www.symantec.com/threatreport/topic.jsp?id=threatreport&aid=malicious_code_trends_report.
- Take the world's best courses, online, for free. (2013). Retrieved May 23, 2013, from <https://www.coursera.org/>
- Talbot, D. (2008, May/June). Where spam is born. *Technology Review*, 111, 28.
- Talbot, David. (2013, June 3). Samsung says new superfast "5G" works with handsets in motion. Retrieved June 4, 2013, from <http://www.technologyreview.com/news/515631/samsung-says-new-superfast-5g-works-with-handsets-in-motion/>
- Tam, Donna. (2013, January 30). Facebook by the numbers: 1.06 billion monthly active users. Retrieved May 20, 2013, from http://news.cnet.com/8301-1023_3-57566550-93/facebook-by-the-numbers-1.06-billion-monthly-active-users/
- Tamim, Z. (2013, August 31). How to fetch and parse JSON using iOS SDK. Retrieved December 12, 2013, from <http://www.appcoda.com/fetch-parse-json-ios-programming-tutorial/>

- Tang, A. & Scoggins, S. (1992). *Open networking with OSI*. Englewood, NJ: Prentice-Hall.
- Tatum, C. (2005). Deconstructing Google bombs: A breach of symbolic power or just a goofy prank? *First Monday*, 10(10).
- Tennant, Roy. (2011, March 17). While web 2.0 (and the related “library 2.0” idea) still receives lots of attention, it has lost some of its “cutting edge” mystique. Retrieved September 27, 2012, from <http://blog.libraryjournal.com/tenantdig/itallibraries/2011/03/17/7-words-or-phrases-to-never-say-or-write-again/>
- Tenopir, C. (2007). Web 2.0: Our cultural downfall? *Library Journal*, 132(20), 36.
- The Text REtrieval Conference. (2007, August 8). Overview. Retrieved May 1, 2008, from <http://trec.nist.gov/overview.html>.
- Top 15 most popular social networking sites. (2013, August). Retrieved August 14, 2013, from <http://www.ebizmba.com/articles/social-networking-websites>.
- Torrone, P. (2011, March 10). Is it time to rebuild & retool public libraries and make “TechShops”? Retrieved May 16, 2014, from <http://makezine.com/2011/03/10/is-it-time-to-rebuild-retool-public-libraries-and-make-techshops/>
- Trend Micro (2008, January). Malware today and mail server security in A *Trend Micro White Paper*. Retrieved May 1, 2008, from <http://www.emediausa.com/FM/GetFile.aspx?id=8541>.
- Trenholm, Rich. (2009, November 2). NSA to store yottabytes in Utah data centre. Retrieved September 24, 2013, from <http://crave.cnet.co.uk/gadgets/nsa-to-store-yottabytes-in-utah-data-centre-49304118/>
- Turkle, S. (2007). Can you hear me now? *Forbes.com*. Retrieved June 4, 2008, from http://www.forbes.com/forbes/2007/0507/176_print.html.
- Turnau, Amber. (2012, July 13). The 5C’s of social media. Retrieved September 17, 2013, from <http://www.6smarketing.com/blog/5-cs-of-social-media-marketing/>
- 250+ Killer digital libraries and archives. (2013, March 25). Retrieved November 7, 2013, from <http://oedb.org/ilibrarian/250-plus-killer-digital-libraries-and-archives/>
- Tyson, Jeff. (2013). Comparing DSL types. Retrieved May 30, 2013, from <http://www.howstuffworks.com/vdsl3.htm>.
- U.S. Census. (2006). Computer and Internet use in the United States: 2003. Retrieved October 1, 2007, from <http://www.census.gov/population/pop-profile/dynamic/Computers.pdf>.
- U.S. Census. (2008, May 15). Quarterly retail e-commerce sales. Retrieved May 22, 2008, from <http://www.census.gov/mrts/www/ecomm.html>.
- U.S. Census. (2010) Computer and Internet use in the United States: 2010. Retrieved April 11, 2013, from <http://www.census.gov/hhes/computer/publications/2010.html>.
- U.S. public libraries weather the storm (2012). Retrieved February 2, 2014, from <http://www.ala.org/research/sites/ala.org.research/files/content/initiatives/plftas/issuesbriefs/issuebrief-weatherstorm.pdf>.
- Unisys. (2008). LZW patent information. Retrieved September 19, 2008, from http://www.unisys.com/about_unisys/lzw.
- Usage of Flash for websites. (2013, July). Retrieved July 16, 2013, from <http://w3techs.com/technologies/details/cp-flash/all/all>.
- Usage statistics and market share of Drupal for websites. (2013). Retrieved July 15, 2013, from <http://w3techs.com/technologies/details/cm-drupal/all/all>.
- Usage statistics and market share of Joomla for websites. (2013). Retrieved September 20, 2013, from <http://w3techs.com/technologies/details/cm-joomla/all/all>.
- Using register globals. (2014). Retrieved January 23, 2014, from <http://www.php.net/manual/en/security.globals.php>.

476 Bibliography

- Valenza, Joyce. (2013, July 15). Qwant a one-page, multilingual search aggregator? Retrieved July 19, 2013, from <http://blogs.slj.com/neverendingsearch/2013/07/15/qwant-a-one-page-multilingual-search-aggregator/>
- van Vuuren, Cairn. (2012, November 18). Mobile apps: The trouble with using 'responsive design'. Retrieved July 1, 2013, from <http://www.forbes.com/sites/ciocentral/2012/11/18/mobile-apps-the-trouble-with-using-responsive-design/>
- Video building block for Blackboard 9.x. (2011). Retrieved July 17, 2013, from <http://exchange.kaltura.com/content/video-building-block-blackboard-9x>.
- Vidmar, D., & Anderson, C. (2002). History of Internet search tools. In A. Kent & C. Hall (Eds.), *Encyclopedia of library and information science* (vol. 71, pp. 146–162). New York: Marcel Dekker, Inc.
- Vlist, Eric van der. (2001, October 17). Using W3C XML Schema. Retrieved August 30, 2013, from <http://www.xml.com/pub/a/2000/11/29/schemas/part1.html>.
- W3C. (1998, October 1). Document object model (DOM) level 1 specification. Retrieved August 30, 2013, from <http://www.w3.org/TR/REC-DOM-Level-1>.
- W3C. (2000, May 8). Simple object access protocol (SOAP) 1.1. Retrieved August 30, 2013 from <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- W3C. (2001). Semantic Web activity. Retrieved October 1, 2007, from <http://www.w3.org/2001/sw>.
- W3C. (2004, February 10). RDF primer. Retrieved December 15, 2013, from <http://www.w3.org/TR/REC-rdf-syntax>.
- W3C. (2005, January 19). Document object model (DOM). Retrieved January 22, 2014, from <http://www.w3.org/DOM>.
- W3C. (2007, July). "WCAG 2.0 Web content accessibility guidelines update." Retrieved May 1, 2008, from <http://www.w3.org/WAI/EO/Drafts/wcag20pres/wcag2intro20070725.doc>.
- W3C. (2010). Scalable vector graphics: XML graphics for the Web. Retrieved December 15, 2013, from <http://www.w3.org/Graphics/SVG>.
- W3C. (2012, April 5). W3C XML schema definition language (XSD) 1.1. Retrieved May 13, 2014, from <http://www.w3.org/TR/xmlschema11-1/>
- W3C. (2013, February 7). Extensible Markup Language (XML) 1.0 (Fifth Edition). Retrieved May 13, 2014, from <http://www.w3.org/TR/REC-xml/>
- W3C. (2013, February 8). The extensible stylesheet language family (XSL). Retrieved December 15, 2013, from <http://www.w3.org/Style/XSL>.
- W3C. (2013, June 27). SemanticWeb. Retrieved May 15, 2014, from <http://www.w3.org/wiki/SemanticWeb>.
- W3C. (2013, December 11). Semantic Web activity. Retrieved December 31, 2013, from <http://www.w3.org/2001/sw>.
- W3C. (2014). Cascading style sheets home page. Retrieved January 22, 2014, from <http://www.w3.org/Style/CSS>.
- W3C working draft. (2011, April 5). HTML5 differences from HTML4. Retrieved June 24, 2013, from <http://www.w3.org/TR/2011/WD-html5-diff-2011-0405/#doctype>.
- W3Schools. (2013). Introduction to XML schema. Retrieved December 15, 2013 from http://www.w3schools.com/schema/schema_intro.asp.
- W3Schools. (2014). CSS tutorial. Retrieved January 22, 2014, from <http://www.w3schools.com/css/default.asp>.
- W3Techs. (2013, July 18). Usage of content languages for websites. Retrieved July 18, 2013, from http://w3techs.com/technologies/overview/content_language/all.
- Walls, Colin. (2006). *Embedded software: The works*. Boston, MA: Elsevier.

- Wang, Avery Li-Chun. (n.d.). An industrial-strength audio search algorithm. Retrieved July 19, 2013 from <http://www.ee.columbia.edu/~dpwe/papers/Wang03-shazam.pdf>.
- Ward, Mark. (2013, June 30). Web porn: Just how much is there? Retrieved July 18, 2013, from <http://www.bbc.co.uk/news/technology-23030090>.
- Ward, E., & French, G. (2013). *Ultimate guide to link building: Build backlinks, earn a higher search engine rank, increase the authority and popularity of your site*. Irvine, CA: Entrepreneur Media.
- Warren, Christina. (2011, June 20). 9 Things you need to know about ICANN's new top level domains. Retrieved August 15, 2013, from <http://mashable.com/2011/06/20/new-gtld-faq/>
- Warren, Samantha. (n.d.). Style tiles: A visual Web design process for clients & the responsive Web. Retrieved July 1, 2013, from <http://styletil.es/>
- Wasserman, Todd. (2011, October 28). Netflix takes up 32.7% of Internet bandwidth. Retrieved September 8, 2013, from <http://www.cnn.com/2011/10/27/tech/web/netflix-internet-bandwidth-mashable>.
- Watkinson, J. (2000). *The art of digital audio* (3rd ed.). Oxford: Focal Press.
- Webber, D., & Peters, A. (2010). *Integrated library systems: Planning, selecting, and implementing*. Santa Barbara, CA: Libraries Unlimited.
- Weber, J. (2006). Evergreen: Your homegrown ILS. *Library Journal*, 131(20), 38–41.
- Webmaster guidelines. (2013). Retrieved December 30, 2013, from <https://support.google.com/webmasters/answer/35769?hl=en>.
- Welcome to fiber cities (2013). Retrieved June 4, 2013, from <https://fiber.google.com/cities/#header=check>.
- Wellman, S. (2007, December 5). Drudge report goes mobile. Retrieved May 19, 2008, from http://www.informationweek.com/blog/main/archives/2007/12/drudge_report_g.html.
- Wells, J., Lewis, L., & Greene, B. (2006). *Internet access in U.S. public schools and classrooms: 1994–2005*. Washington, D.C.: U.S. Department of Education.
- Wenzl, Roy. (2013, January 19). Billions of sensors power Wichita State professor's vision of interconnected world. Retrieved January 25, 2013, from <http://www.kansas.com/2013/01/19/2643201/billions-of-sensors-power-wichita.html#storylink=misearch>.
- What is Java technology and why do I need it? (2013). Retrieved May 23, 2013, from http://www.java.com/en/download/faq/whatis_java.xml.
- What is Joomla? (2013). Retrieved July 15, 2013, from <http://www.joomla.org/about-joomla.html>.
- White, M. (2000, February 2). Loans.com latest web name to make millionaire of seller. *Herald-Leader*, p. C2.
- Whitwam, Ryan. (2013, June 7). Kaspersky researchers discover most advanced Android malware yet. Retrieved June 17, 2013 from <http://www.androidpolice.com/2013/06/07/kaspersky-researchers-discover-most-advanced-android-malware-yet/>
- Who provides Internet service for my Internet service provider? (2013). Retrieved May 29, 2013, from <http://www.howtogeek.com/123599/who-provides-internet-service-for-my-internet-service-provider/>
- Williams, A. (2013). *SEO 2013 and beyond: Search engine optimization will never be the same again!* CreateSpace Independent Publishing Platform.
- Wilson, T. D. (2000). Human information behavior. *Informing Science*, 3(2), 49–56.
- Windows rootkit overview. (2006, March). Retrieved January 31, 2014, from <http://www.symantec.com/avcenter/reference/windows.rootkit.overview.pdf>.
- Wisniewski, J. (2008). The new rules of Web design. *Online*, 32(2), 55–57.
- Wolf, G. (1995, June). The curse of Xanadu. *Wired*, 3, 137–202.

478 Bibliography

- Wong, Q. (2008, May 1). WiMax to widen Net access: New device more powerful, secure than Wi-Fi. *Lexington Herald-Leader*, p. A3.
- WordPress sites in the world. (2013). Retrieved September 26, 2013, from <http://en.wordpress.com/stats/>
- Wrolstad, J. (2004, September 10). FCC: Broadband usage has tripled. Retrieved May, 2007, from http://www.newsfactor.com/story.xhtml?story_title=FCC—Broadband-Usage-Has-Tripled&story_id=26876.
- Wyatt, Edward. (2010, April 6). U.S. court curbs F.C.C. authority on Web traffic. Retrieved April 13, 2013, from http://www.nytimes.com/2010/04/07/technology/07net.html?pagewanted=all&_r=0.
- Wyatt, Edward. (2014, April 23). FCC will allow Internet fast lane, sidestepping 'Net Neutrality'. Retrieved April 24, 2014, from <https://www.yahoo.com/tech/fcc-will-allow-internet-fast-lane-sidestepping-net-83658252368.html>.
- Yaneza, J.L., Thiemann, T., Drake, C., Oliver, J., Sancho, D., Hacquebord, F., et al. (2008). 2007 Threat report and 2008 threat and technology forecast. Retrieved June 18, 2013, from http://us.trendmicro.com/imperia/md/content/us/pdf/threats/securitylibrary/tre_threat_reprep.pdf.
- Yates, Ian. (2013, May 13). Photoshop's role in a Web design workflow. Retrieved July 1, 2013, from <http://psd.tutsplus.com/articles/tools/photoshop-role-in-web-design/>
- Yonatan, R. (2013, August 13). 35 Powerful Cloud Tools for Modern Librarians. Retrieved October 25, 2013, from <http://getvoip.com/blog/2013/08/13/35-powerful-cloud-tools-for-modern-librarians>.
- Yott, P. (2005). Introduction to XML. *Cataloging & Quarterly*, 40(3/4), 213–235.
- Zaino, J. (2013, May 15). Talking the talk—And walking the walk—About the beauty of search at Google I/O. Retrieved January 4, 2014, from <https://semanticweb.com/tag/amit-singhal>.
- Zakon, R. (2011). Hobbes' Internet timeline. Retrieved April 11, 2013, from <http://www.zakon.org/robert/internet/timeline/>
- Zeitgeist overview. (2013). Retrieved September 6, 2013, from <http://www.libr arything.com/zeitgeist>.
- Zeller, T., Jr. (2005, February 1). Law barring junk e-mail allows a flood instead. *New York Times*, p. A1.
- Zickuhr, Kathryn. (2010, December 16). Generations online in 2010. Retrieved June 11, 2013, from http://pewinternet.org/~/media/Files/Reports/2010/PIP_Generations_and_Tech10_final.pdf.
- Zillman, Marcus P. (2006, June 10). Features—Deep Web research 2006. Retrieved July 15, 2013, from <http://www.llrx.com/features/deepweb2006.htm>.
- Zimmer, M. (2008). Preface: Critical perspectives on Web 2.0. *First Monday*, 13(3).
- Zipf, G. K. (1949). *Human behavior and the principle of least effort; an introduction to human ecology*. Cambridge, MA: Addison-Wesley.
- Zittrain, Jonathan. (2008). *The future of the Internet and how to stop it*. New Haven, CT: Yale University Press.
- Zukerman, Erez. (2012, April 9). Create a Website easily with Wix (even the free version). Retrieved July 11, 2013, from <http://www.pcworld.com/article/253379/wix.html>.

Index

- Absolute reference, 104, 184–87
Abundance problem, 363, 376
Accessibility: authentication, and, 127; browser, and, 36; color, and, 157; design, and, 28, 136–37, 167; framesets, and, 193; HTML5, and, 201; images, and, 158, 165, 182; mobile, and, 28; scripting, and 153, 235, 242; standards, 166–68, 323; tables, and, 151, 187; testing, and, 167
Active Server Pages (ASP), 137, 229, 312
ADA. *See* Americans with Disabilities Act
Addressing granularity, 330
Address resolution protocol (ARP), 47, 54, 74, 76, 81, 83, 86, 92, 120
Adobe: Acrobat, 310–11; CQ5 Communi- que 5, 262; Dreamweaver, 144, 255; Fireworks, 149; Portable Document Format, 115, 310–12; Photoshop, 149, 162–63, 212; Postscript, 311
ADSL. *See* Asymmetric Digital Subscriber Line
Advanced Research Project Agency Network (ARPANET), 3, 5, 6, 69, 70
Advertising: cookies, and, 121; reve- nues, 38; social media, and, 30, 38; search engines, and, 141, 357; spy- ware, and, 121
AI. *See* Artificial intelligence
AIP. *See* Archival Information Package
AJAX. *See* Asynchronous JavaScript and XML
Aldus Tagged Image (TIFF), 164
Algorithm: 228, images and, 164–65; IR, and, 339, 350, 360, 362–63, 367–70
Aliasing, 12, 157, 160–61
ALIWEB. *See* Archie-Like Index of the Web
Amazon: 37–38; APIs, 411; cloud services, 16, 395, 414; Elastic Compute Cloud (EC2), 31, 35; eBooks, 312–13; libraries, and, 400–401; Prime, 317; search, and, 377
American Standard Code for Information Interchange (ASCII): 24, 26, 425, 427–28; character entities, 177; email, and, 97; files, 154, 308, 309, 312; FTP, and, 100; HTML, and 176; HTTP, and, 103, 104; scripts, and, 235; XML, and, 282
Americans with Disabilities Act (ADA), 166
Amplitude modulation, 11
AMV video, 316
Analog: aliasing, and, 160–61; band- width, and, 45; computers, 10, 12; conversion to digital, 12, 56; 156,

- 316; data, 11, 64; multimedia, 315; telephone service, 51, 56, 57, 59, 66, 100
- Android: 15, 32, 63, 109; design for, 155, 318; Market, 231; security, and, 110, 111, 122
- Animations: CSS3, and, 215; Flash, 140, 318; GIF, 164
- Anti-aliasing, 160–61
- Antivirus, 117, 120, 123, 124, 127–129, 320
- API. *See* Application programming interface
- APIPA. *See* Automatic private IP addressing
- Apollo ILS, 399
- App: 15, 252, 253, 396; APIs, and, 411; cookies, and, 378; examples, 29, 32, 145, 146, 200, 342, 378, 379, 406; HTML5, and, 200; HTTP, and, 105; native, 231; programming, 231; security, and, 122; store, 122, 231
- Apple: BinHex, and, 320; computers, 9, 31, 162, 169, 318, 396; eBooks, 313; iCloud, 16; 33, 35; iOS, 32, 122, 318; iPad, 7, 15, 159; iPhone, 15, 62; iPod, 364, 409; iTunes, 35, 38, 39, 231, 315; iTunes U, 39; QuickTime, 316, 317; retina display, 159; security, and, 109–111, 113, 122, 125; Siri, 331, 365, 372, 382; Stuffit, 320; voice search, and, 372
- Applet: defined, 120; tag, 234
- Application Programming Interface (API), 35, 44, 46, 63, 91, 200, 243, 297, 412
- Archie index, 100, 101, 352
- Archie-Like Index of the Web (ALIWEB), 353
- Archival Information Package (AIP), 269
- ArchivesSpace, 270
- Argument: end-to-end, 70; in commands, 228; in functions, 245
- ARP command, 74
- ARP. *See* Address Resolution Protocol
- ARPANET. *See* Advanced Research Project Agency Network
- Artificial intelligence (AI), 364–65, 393
- ASCII. *See* American Standard Code for Information Interchange
- ASP. *See* Active Server Pages
- Assistive Technology, 167
- Assistive Technology Act, 166
- Asymmetric Digital Subscriber Line (ADSL), 59
- Asynchronous JavaScript and XML (AJAX): 27, 297, 300–301, 303, 412; in design, 145; security, and, 113; widgets, and, 242, 252
- AT&T, 8, 56, 58, 60, 65
- Audio Visual Interleave (AVI), 316, 317, 319
- Authentication: invisible web, and, 27, 360; IP, and, 76; library systems, and, 399, 402; passwords, and, 125; proxy servers, and, 55, 87; TCP, and, 107; security, and, 109, 114, 115, 127; VPNs, and, 54
- Automatic indexing, 332, 353
- Automatic private IP addressing (APIPA), 79, 82
- Avatar, 393, 394
- AVI. *See* Audio visual interleave
- Babbage, Charles, 10
- Backdoor app, 122
- Backlink, 368
- Bandwidth: 3, 13, 17, 19, 28, 67, 390; analog, 47, 56; defined, 14–15, 45; Internet connections, and, 53, 55, 57, 58, 60, 61; mobile, 62, 64, 66, 145; multimedia, and, 316–318; net neutrality and, 8, 9; sampling rates, and, 12; streaming, and, 75, 99, 317; virtual reality, and 393
- Banner ads, 357, 361
- Base 16 numbers. *See* Hexadecimal numbering
- Baseband signal, 53
- Base tag, 184, 196
- Basic Input/Output System (BIOS), 44, 91
- Baud rate, 56
- Bell Laboratories, 12
- Berners-Lee, Tim, 6, 23, 24, 25, 101, 175, 275, 276, 351, 352, 370, 393, 395
- Big data, 16, 348, 365, 372, 381
- Big-S Semantic Web, 276, 395
- Binary files, 97, 230, 275, 308–9, 311, 320, 425
- Binary numbers: computers, and, 3, 12, 13; explained, 425–28; in color schemes, 161–62; in packet addressing, 54, 76–78
- Bing smart search, 375

- BIOS. *See* Basic Input/Output System
- Bitmap images. *See* Raster image
- BITNET, 96
- Bit resolution, 163
- Blackboard Learning Management System, 37, 39, 80, 256, 265, 266, 315, 317
- Block-level elements, 180–81, 199–200, 206
- Blogs: as searchable content, 307, 314, 321, 323; defined, 29; demographics of, 98, 403, 407; examples, 38, 144, 262, 263, 340; history, 7, 17; in CMSs, 264, 265; in libraries, 98, 400, 409, 410; in LMSs, 39, 266; RSS, and, 297; security, and 110; social signal, and, 372; Web 2.0, and 27, 313, 388, 389, 391, 392
- Bluetooth, 50
- BOM. *See* Browser Object Model
- Bookmark: in design, 141; in HTML, 183; sharing, 376, 407, 410–12
- Boolean logic: explained, 428–29; fuzzy, 335, 337, 356, 362, 373, 381, 401; in computing, 3, 12; in CSS, 217; in XML, 285; IR Model, 335–37; subnet masks, and, 85–86; in search, 352, 355, 356, 361, 379, 400; script operators, 236, 244
- Boopsie, 412
- BOOTP, 81–82
- Botnet: defined, 121; Pony, 126; Rustock, 96, 121
- Bots: 96; defined, 121; CAPTCHA, and, 127
- Box Model, 198, 205–6, 215
- BPL. *See* Broadband over power lines
- Branding, in design, 136, 138, 141, 266
- Breadth-first indexing, 359
- Bridge, 47, 81
- Brin, Sergey, 362, 364
- Bring your own device (BYOD), 32
- Broadband: cable, 59–60; digital divide, and, 65–66; DSL, 14, 59; fiber, 60; ISDN, 58; net neutrality, and, 9; over power lines, 61; security, and, 110; services, 55, 57, 66–67; satellite, 60; streaming, and, 98, 99, 317; Web, and, 390, 395; Wi-Fi, 61, 65
- Broadband over power lines (BPL), 61
- Browser: as a platform, 16, 33, 34, 35, 40, 97, 203, 317, 395; color rendering, 162; cookies, 105; CSS, and, 206–7, 213, 215–16; extensions, 36–37, 233; history, 6, 25, 203, 205, 352; HTML, and, 175, 176, 178–80, 182, 197, HTML5, and, 200; HTTP, and, 26, 90, 102, 104, 106, 184; images, and, 155, 165, 310; in design, 135, 141, 149, 153, 154, 169; mobile, and, 145, 147, 231; modes, 37; processing, 158, 193, 233–34, 240; security, and, 113, 114, 126, 127; XML, and, 293–94, 302
- Browser Object Model (BOM), 238, 239
- Bush, Vannevar, 10, 24
- Butcher paper method, in design, 147
- BYOD. *See* Bring your own device
- Byte boundary mask, 84
- California Digital library Curation Center (UC3), 270
- Canvas, 40, 266
- CAPTCHA, 127
- Carrier Sense, Multiple Access with Collision Detection (CSMA/CD), 53
- Cascading Style Sheets (CSS): !important designation, 214, 216; box model, 198, 205–6, 216; cascade, 204, 213–14; classes, 209–11; color control, 212–13; CSS3 modules, and, 214; declarations, 208–14, 216, 217; defined, 204–5; DIV containers, 206, 209, 214; embedded styles, 208; examples, 209–13, 220–24; external files, 207; hacks, 215–16; history, 203–5; HTML5, and, 214; inline styles, 208; positioning, 206; pseudo-classes, 211; responsive design, and, 216–19; selectors, 206, 208–11, 213, 214, 217, 294; shortcuts, 216; sizing, 211–12; style tag, 206–7; syntax and rules, 208–9; with XML, 294–95
- Catalog, 399–402
- CBIR. *See* Content-Based Information Retrieval
- CCM. *See* Component content management
- CDROM: databases, 24; music, 315
- Cellular data services: 28, 51, 62; 3G/4G LTE, 15, 21, 54, 64–66
- Centurylink, 56
- Cerf, Vinton, 58, 70
- CERN, 6, 23, 24, 175, 352
- CGI. *See* Common Gateway Interface

- Chromebook, 34, 36, 364
 Chrome browser, 25, 37
 Chromium OS, 36
 Circuit switching, 51, 57
 Citation analysis, 362, 369
 Citrix client/receiver, 32
 CKEditor, 144, 260
 Class2Go, 40
 Clickable images, 192
 Click through model, 357
 Client-server architecture, 13, 34, 95, 112
 Client-side scripting, 233–35
 Cloud: APIs, 200; architecture, 34; CMSs, 262–63, 265, 271; computing, 33, 394–95; defined, 16; examples, 31–32, 35–36, 38, 97, 163, 364, 415; library systems, 388, 396, 399; mobile, and, 412; tradeoffs, 417; virtualization, and, 30–32
 Clustering results, 376–77
 CMA. *See* Content management application
 CMS. *See* Content management system
 CMYK color, 161, 162
 Coaxial cable, 45, 48–50, 53
 Color: cultural effects, and, 157; design, and, 137, 141, 157–58; file formats, and, 163–65; HTML, and, 161–62; in CBIR, 342; in CSS, 213; models/schemes, 161–62
 Color depth, 159, 161, 162, 170
 Comcast, 9, 58
 Commands: arp, 74; ipconfig, 92; netstat, 92, 127–129; ping, 76, 87, 92, 128; traceroute, 76, 92; whois, 88
 Common Gateway Interface (CGI), 114, 229
 Comp, in design, 136, 149, 212, 218
 Compiled programs, 229–30
 Component content management (CCM), 256, 271
 Comprehensive search, 343, 356, 358
 Compression: file formats, 310, 316–318; lossless, 164; lossy, 165, 319; PPPoE, and, 81 LZW, 164
 Computers: analog, 10, 12; digital, 12; personal, 9; trends, 13–17
 Content-Based Information Retrieval (CBIR), 340–43, 379
 ContentDM, 270, 398
 Content management application (CMA), 362
 Content management system (CMS): defined, 255; types, 255–56; examples, 262–66; technologies, 260–62
 Controlled vocabularies, 330–33, 337, 341, 342, 348, 351, 379, 380, 403
 Control node, 33
 Cookies: 26; defined, 105–6; in Flash, 115, 121; in HTML5, 200; in scripting, 227, 252; mobile, and, 377–78; security, and, 113, 120–21
 Coursera, 40
 Crawler programs, 7, 350, 353, 359, 360, 364–367
 Cross domain scripting, 113, 301
 Crossover cable, 48
 CSMA/CD. *See* Carrier Sense, Multiple Access with Collision Detection
 CSS. *See* Cascading Style Sheets
 CSS3 modules, 204, 214–15
 Cunningham, Ward, 29
 DAD. *See* Duplicate Address Detection
 Daemon, 26, 112, 261
 Damping factor, PageRank, 367–68
 Darwin Information Typing Architecture (DITA), 256
 Database: design, 258; relational, 256–59; SQL, and, 257, 259; types, 257; XML, and, 275, 277, 280; Web programming, and, 230–31
 Database Management System (DBMS), 257
 Database of intentions, Google, 347
 Data bus, 12
 Dataflow Diagrams, 138, 147
 Datagrams. *See* Packets
 DBMS. *See* Database Management System
 Deep Web. *See* Invisible Web
 Delicious, 411, 412
 Denial of service (DoS), 74, 113, 114, 116
 Deprecated tags, 175, 180, 183, 193, 299
 Depth-first indexing, 359
 Descriptive (or generic) markup, 174, 178–79
 Deutsche Telekom, 56
 DHCP. *See* Dynamic Host Configuration Protocol
 Dialog, 379, 380, 382, 390
 Dial-up, 55–56, 59, 66, 81, 99, 379

- Digital: conversion from analog, 12, 160, 315; data representation, 3, 12, 161; sampling, 12, 156, 315–16
- Digital archives, 266–69
- Digital commons, 270
- Digital divide, 55, 65–66, 97
- Digital repositories, 256, 266–69
- Digital Subscriber Line (DSL), 14, 55, 59
- DIP. *See* Dissemination Information Package
- Directory search services: 343, 353–54; subject, 152, 411
- Directory tree: in menus, 354; naming, 142; permissions for, 432–33; references in, 184–87, 196
- Discovery tools, 381, 400, 402–3
- Discussion lists, 97–98, 313, 314, 389, 409
- Dissemination Information Package (DIP), 269
- DITA. *See* Darwin Information Typing Architecture
- Dithering, 162
- DMOZ, 354
- DNS. *See* Domain Name System
- DNS poisoning, 120
- Document Object Model (DOM): 180, 199, 200, 238, 241, 243; in XML, 296, 300, 303
- Document processing, 329–30, 332
- Documents-to-terms matrix, 338
- Document Type Definition (DTD): HTML Frameset, 178; HTML specification, 175–77; HTML Strict and Transitional, 175; SGML, and, 134–35; validity, and, 277, 282–83; XHTML, and, 299; XML, and, 283–86
- Dojo Toolkit, 243
- DOM. *See* Document Object Model
- Domain names: fully qualified, 25, 90; registering, 88–90; suffixes, 88
- Domain Name System (DNS): overview, 7, 87; DNS lookup, 90–91
- Doodle polls, 36, 414
- DoS. *See* Denial of service
- Dot notation, 210, 238
- Dot pitch, 159, 163, 211
- Dots per Inch (DPI), 163
- Dotted quad, 77
- DPI. *See* Dots per Inch
- Drive by attack, 110
- Drivers (software), 91, 100
- Dropbox, 33, 35, 414
- Drupal, 17, 136, 144, 231, 259, 260, 262, 264–65
- DSL. *See* Digital Subscriber Line
- Dspace, 270
- DTD. *See* Document Type Definition
- Dublin Core, 293, 298
- Duplicate Address Detection (DAD), 82
- Dynamic Host Configuration Protocol (DHCP), 79, 81–83
- eBay, 39, 377
- eBooks: 36, 138; formats, 311–13; in libraries, 398, 401–3, 412, 417; sales of, 39
- EbscoHost, 139, 380, 381
- EC2. *See* Elastic Compute Cloud
- ECMAScript. *See* JavaScript
- eCommerce, 7, 16, 37–38
- Elastic Compute Cloud (EC2), 31, 35
- Electromagnetic field (EMF), 49
- Elk Cloner virus, 111
- Email: as searchable content, 313–14; HTML, and, 188, 190–92; MIME, and, 320; providers, 34, 142, 355, 364, 414; in discussion lists, 28, 313, 314; protocols, 96–98, 107, 320; reference, 409; security, and, 113, 116–20, 122, 129, 259; spam, and, 121; use of, 19, 98, 113, 347, 351, 413
- Embedded styles, 208
- EMF. *See* Electromagnetic field
- Empty elements, 165, 177, 179–82, 190, 197, 279, 282, 299
- Encapsulation, packet, 46, 72
- Encoding schemes, 13, 97, 154, 309, 320, 425
- Entity Relationship Diagrams, 138, 258, 298
- E-rate, 66
- ERIC: database, 333; descriptors, 333; thesaurus, 331, 333–34
- Ethernet: addresses, 46, 54; frame, 52–54, 72, 86–87; port, 46–48, 59, 61, 81; speeds, 53; standards, 52–53; topologies, 48; wiring, 48, 49, 53
- Ethernet II, 52–53
- Evaluation rubrics: CARS, 322; five W's, 322
- Evergreen ILS, 399, 401, 402
- Existence search, 343, 355, 356

- Exploratory search, 331, 333, 343, 353, 378
- Extended ASCII, 308, 428
- Extensible Markup Language. *See* XML
- Extensible Stylesheet Language. *See* XSL
- Eyejot, 36, 414
- EZProxy, 55, 398
- Fab labs, 404
- Facebook: critics of, 416; in libraries, 406, 407; JSON, and, 300; security, and, 114, 118, 121; search, and, 379, 405; use and users, 17, 30, 339, 408, 410; Web 2.0, and, 27, 29, 36
- Fake sites, 120
- False drops (or false hits), 326, 332
- FCC. *See* Federal Communications Commission
- Federal Communications Commission (FCC), 9, 56, 58, 61, 65
- Federated search, 27, 138, 400, 401–3
- Fiber cable, 53, 60–61
- File permissions, 99, 100, 432–33
- File Transfer Protocol (FTP), 13, 28, 95, 100, 101, 111, 123, 127, 128, 137, 351–52, 433
- Findability, 168, 350
- FiOS, 60
- Firewalls: invisible Web, and, 27, 350; ping, and, 76, 92, 128; private networks, and, 54, 114; proxy servers, and, 54, 87; security, and, 112–13, 124, 129; VMs, and, 31
- Flash cookie, 115, 121
- Flash file format (SWF), 164, 318
- Flat file databases, 257
- Flexible images, 146
- Flickr, 35, 36, 144, 300, 301, 377, 389, 409–11
- Floating frames. *See* Iframes
- Flow content, 200, 206
- Fluid grids, 146
- Folksonomies, 331, 340, 342, 403
- Font: alignment, 155; anti-aliasing, and, 160–61; baseline, 154; CSS, and, 203, 211–12; family, 154; leading, 155; serif/sans serif, 155; sizing, 154; typeface, 154
- Forms: in HTML, 187–92, 195, 199, 229, 248–49, 311; in search, 257, 259, 361
- Foursquare, 18, 406, 408
- FQDN. *See* Fully Qualified Domain Name
- Frames: in framesets, 151, 168, 175, 178, 193–94, 238; link control, and, 196; mobile, and, 141, 198; search, and, 359
- FRBR. *See* Functional Requirements for Bibliographic Records
- Frequency modulation, 11
- Frontier Communication, 60
- FTP. *See* File Transfer Protocol
- Fully Qualified Domain Name (FQDN), 25, 90
- Functional Requirements for Bibliographic Records (FRBR), 396
- Fuzzy Boolean, 335, 337, 357, 362, 373, 381, 401
- Garfield, Eugene, 362
- Gateway, 52, 75, 86–87
- General Markup Language (GML), 174, 277
- Generic markup. *See* Descriptive markup
- Generic Top-Level Domain (gTLD), 7, 88
- Geolocation, 15 145–47, 229, 231, 377, 407, 413
- GIF. *See* Graphics Interchange Format
- GML. *See* General Markup Language
- Gnutella, 375–76
- Google: adwords, 357; AI, and, 364–65; analytics, 16, 136, 169, 170, 405; apps, 33, 414; architecture 365–66; Bigtable, 366; Books, 39, 355, 397, 400; carousel, 374; Chrome, 25, 36–37; Chromebook, 34, 36, 364; Chromium OS, 36; Drive, 16, 17, 33–35, 188, 195, 256, 414; Feedburner, 144; fiber, 60; file system, 365; Hangouts, 364; history, 364–65; image viewer, 269; image search, 340, 379; Knowledge Graph, 356, 373–74; map API, 243, 412; PageRank, 339, 362, 364, 366–69; Panda, 369–70; Penguin, 370; Play 15, 231; revisions, 369–70; Scholar 270, 348, 355, 381, 397; semantic search, 370–72; Sites, 262–63; snippets, 374–75
- Google fiber, 60–61
- Google generation, 397
- Googlization, 364, 397–98, 402
- Gopher, 23, 25, 101, 137, 352, 389

- Graphical user interface (GUI), 158, 229, 389
- Graphics Interchange Format (GIF), 155, 162, 164, 203, 310
- Grids, 30, 32–33, 34
- Griefer, 394
- gTLD. *See* Generic Top-Level Domain
- GUI. *See* Graphical user interface
- Hackers, 51, 111–12, 114, 120, 121, 123
- Hackerspaces, 404
- Hardware as a service (HaaS), 395
- Harmony client, 203
- Harvester, 120, 360
- Hashtag, 29, 408–9
- Heartbleed bug, 109, 112, 114–15
- Hexadecimal numbering, 54, 76, 77, 157, 162, 213, 320, 427
- High Performance Computing Act, 6
- HITS Model. *See* Hypertext Induced Topic Search
- Hoaxes, 120, 128, 129
- Homonymy, 332–33, 337
- Honeypot, 120
- Horizontal resolution, 159
- Hotspot (Wi-Fi): 51, 62, 66, 110; security, and, 122
- HREF. *See* Hypertext reference
- HSL. *See* Hue, Saturation, Lightness Model
- HTML. *See* Hypertext Markup Language
- HTML 4.01, 174–77
- HTML5: accessibility, and, 201; app programming, and, 231; browser support, and, 200; CSS, and, 206; CSS3 modules, and, 214–15, 225; development, 198–99; eBooks, and, 312; embedded metadata, and, 298, 371; HTML 4.01, and, 175–77, 180, 181, 183, 193, 197, 199–200; mobile, and, 105, 197, 198, 262; multimedia, and, 165; new features of, 200; scripting, and, 228; standard, 174; XML, and, 275, 278
- HTTP. *See* Hypertext Transfer Protocol
- HTTPd. *See* Hypertext Transfer Protocol Daemon
- HTTPs. *See* Hypertext Transfer Protocol secure
- Hub: HITS model, 363; network, 45–46
- Hue, Saturation, Lightness Model (HSL), 161–62
- Hypertext: CDROM, 24; in design, 153; in Web development, 23–24; Xanadu, and, 24
- Hypertext Induced Topic Search (HITS), 362–63
- Hypertext link analysis, 26, 168, 334, 362–64, 366, 367, 369, 376, 381
- Hypertext Markup Language (HTML): attributes and elements, 179–80; block-level and inline elements, 180–81; common tags, 181–82; DIV containers, 181, 206; history, 23, 25, 175–76; images, in, 182–83; specification, 176–78
- Hypertext reference (HREF): absolute, 104; relative vs. absolute, 184–87
- Hypertext Transfer Protocol (HTTP): development, 23–24; keep alive feature, 106; methods and versions, 101–4; pipelining, 106; secure, 106–7; security, and, 112–13; statelessness, and, 105
- Hypertext Transfer Protocol Daemon, 26, 112
- Hypervisor; rootkits, and, 117; VMs, and, 31
- IaaS. *See* Infrastructure as a service
- IAB. *See* Internet Architecture Board
- IANA. *See* Internet Assigned Numbers Authority
- IBM Mainframe, 20, 99
- ICANN. *See* Internet Corporation for Assigned Names and Numbers
- ICMP. *See* Internet Control Message Protocol
- Iconography, 158, 377
- Ideographs, 158
- IEEE 802.3, 53
- IETF. *See* Internet Engineering Task Force
- Iframes, 194–95, 201; links, and, 196; mobile, and, 147, 198; security, and, 113; XHTML, and, 299
- ILLiad, 398
- ILS. *See* Integrated Library System
- IM. *See* Instant Messenger
- Images: formats, 163–65, 310; HTML, and, 182–83; maps, 192–93, 193; searching for, 339–43, 360; symbolism, and, 158
- IMAP. *See* Internet Message Access Pool
- IMP. *See* Interface Message Processor

- Include files, 233
- In-degree/out-degree, 363
- Indexical bias, 358
- Indexing: automatic, 332; depth, 330; function of, 328–30; human, 331–32
- Index overlap. *See* Indexical bias
- Info Archipelago, 394
- Information: definitions, 326; scent, 142; seeking behavior, 343–44
- Information literacy, 138–39, 307, 320–23, 398, 416
- Information retrieval (IR): definitions, 325–26; models, 335–39; multimedia, 339–43; textual, 327–28
- Infrastructure as a service (IaaS), 34
- Inline elements, 180–81, 200, 206
- Inline frame. *See* Iframe
- Instagram, 30, 35, 406–8; security, and, 114
- Instance document, 175, 178, 281–84; examples of, 285, 287, 290; in JSON, 301; XSD, and, 287, 289
- Instant messaging (IM), 36, 99–100, 313, 389, 396, 408–9, 413, 416
- Integrated Library System (ILS): defined, 398–99; next generation, 401; options, 399
- Integrated Services Digital Network (ISDN), 55, 57, 58
- Interface Message Processor (IMP), 5
- Internet: defined, 5; history, 5–7; of everything, 17–19, 77, 417; of information, 18, 307; of people, 18, 28; of things; 18–19
- Internet administration, 7–8, 87–90
- Internet Architecture Board (IAB), 5, 8
- Internet Assigned Numbers Authority (IANA), 8, 88
- Internet Control Message Protocol (ICMP), 74–76, 87
- Internet Corporation for Assigned Names and Numbers (ICANN), 7–8, 88
- Internet Engineering Task Force (IETF), 8, 25, 106
- Internet exchange point (IXP), 8
- Internet Message Access Pool (IMAP), 96–97, 107
- Internet Protocol (IP): address headers, 71–73; packets, 69–71
- Internet Protocol v6 (IPv6), 7, 69, 72, 80, 93; ARP, and, 74; datagrams, 72–73; DHCP, and, 82–83; DNS, and, 90; ICMPv6, and, 76; NAT, and, 79; packet addressing, 73, 77
- Internet Relay Chat (IRC), 99–100, 111; security, and, 123, 127–28
- Internet Service Provider (ISP): 55–56; address space, 83; charge models and metering, 58; CMS, and, 262; net neutrality, and, 9
- Internet Society, 7, 80
- Internet2, 6–7
- Internetwork Packet Exchange/ Sequenced Packet Exchange (IPX/SPX), 52
- Interpreted programs, 229
- Intranets, 5, 52, 78, 347
- Inversion, 329–30
- Inverted index (also inverted file or inverted file index), 328–30
- Invisible Web (also deep Web), 26–27, 40, 227, 308; searching, and, 348, 350–51, 360, 378
- iOS, 32, 122, 318
- iPad, 7, 15, 159
- IP address assignment: BOOTP, 81–82; DHCP, 82–83; hard coding, 81; RARP, 81–82
- IP addressing: classes, 77–78; dotted quads, 77; octets, 77
- Ipconfig command, 92
- iPhone, 15, 62
- IPng. *See* Next Generation IP
- iPod, 364, 409
- IPv6. *See* Internet Protocol v6
- IR. *See* Information retrieval
- IRC. *See* Internet Relay Chat
- ISDN. *See* Integrated Services Digital Network
- ISP. *See* Internet Service Provider
- iTunes, 35, 38, 39, 231, 315
- iTunes U, 39
- IXP. *See* Internet exchange point
- Java programming, 230, 234, 297
- JavaScript: ADA, and, 153, 167; apps, and, 231; basics, 237–39; cookies, and, 106; defined, 238; history, 237–38; examples, 236, 239–43; functions, 237, 240–42; include files, and, 233; Java, and, 230; objects, 238; operators, 237; search engines, and, 168; security, and, 113, 115, 116; testing, 37, 233; tools, 233; variables, 236

- JavaScript console, 37, 233
 JavaScript Object Notation (JSON), 113, 276, 300–303, 389
 Java Server Pages (JSP), 230, 234, 256, 258
 Java Virtual Machine (JVM), 32, 230, 234
 Joint Photographic Experts Group format (JPG or JPEG), 155, 164, 165, 310
 Journal impact factor, 362
 JPEG/JPG. *See* Joint Photographic Experts Group format
 JQuery, 37, 228, 243, 264, 300, 301
 JSON. *See* JavaScript Object Notation
 JSP. *See* Java Server Pages
 JVM. *See* Java Virtual Machine
- Kahn, Robert, 70
 Kaltura, 317, 323
 Kartoo, 377
 Kentucky Digital Library (KYVL), 270
 Kernel, 31, 117
 Killer app, 96
 KISS rule, 142
 Kleinberg, Jon, 362–63
 KLEZ worm, 116
 Known item search, 343
 Koha ILS, 399, 401
- LAMP, 31, 233, 257, 262
 LAN. *See* Local Area Network
 Landline service, 57, 60, 65, 66
 Layout, in Web design, 146, 147, 149–51
 LDAP. *See* Lightweight Directory Access Protocol
 Leading, fonts, 155
 Learning management system (LMS), 37, 80, 265, 266, 271
 Level 3 Communications, 56
 Lexis-Nexis, 356, 379, 381
 LibGuides, 142, 265–66, 411
 Library 2.0, 388, 390, 392, 396, 409, 415
 LibraryThing, 400
 Lightweight Directory Access Protocol (LDAP), 107
 Linden Labs, 393
 Link analysis, 168, 334, 362–63, 364, 381; in PageRank, 367, 369
 Link bombs, 350, 369
 LinkedIn, 18, 27, 29–30, 406–8
- Link resolver, 26, 141, 260, 398, 402
 Linux, 9–10, 17, 31, 114, 233, 257, 262, 431
 LISTSERV, 98, 313
 LMS. *See* Learning management system
 Local Area Network (LAN), 44, 46, 48, 54, 66, 79, 82, 399
 Logical markup, 178–79
 The long tail, 17–18
 Loopback address, 79, 92
 Lossless compression, 164
 Lossy compression, 164–65, 317, 319
 LZW compression, 164
- M2M. *See* Machine-to-machine
 MAC. *See* Media access control
 Machine-to-machine (M2M), 33
 Macro viruses, 115
 MailChimp, 36, 414
 MajorDomo, 313
 Maker culture, 404, 418
 Maker spaces, 404
 Malicious Mobile Code (MMC), 110, 115, 116
 Malware, 19, 97, 105, 110, 320; mobile, and, 111, 122; types, 115–17, 121, 122
 MAMP, 262
 MARC, 71, 332
 Marcotte, Ethan, 146, 150, 212, 219
 Markup languages, 26, 174
 Mashups, 7, 27, 144, 243, 388–89, 391, 411–13
 Massive, Multiplayer, Online Role-Playing Game (MMORPG), 393
 Massive Open Online Course (MOOC), 40
 McCahill, Mark, 101
 Media access control (MAC), 46
 Media queries, 146, 149, 216–17
 Memex, 10, 24
 Merritt repository service, 270
 Mesh, 48–49
 Metadata: anchor text, and, 367; defined, 327–28; folksonomy, and, 331, 340, 342, 403; HTML, and, 181; HTTP, and, 102; interoperability, 276–77, 297; searching, and, 327, 351, 360–62, 366, 370–72, 374–75, 381; standards, 269, 271, 293, 297, 303; XML, and, 275, 278, 282, 293

- Metadata Encoding and Transmission Standard (METS), 297–98, 389
- Metadata Object Description Schema (MODS), 297, 303, 389
- Metasearch: in libraries, 27, 138, 403; search engines, 343–44, 349, 353, 356, 358
- METS. *See* Metadata Encoding and Transmission Standard
- Microdata, 351, 361, 366, 371–72, 374–76
- Microformats, 371–72, 374
- Microservice architecture (MSA), 269–70
- Microsoft: .NET initiative, 229; ASP (Active Server Page), 137, 229, 232, 234, 261, 312, 431; Bing, 353, 357, 364, 371, 372, 375; Exchange Server, 97; Hotmail, 98; Internet Information Services (IIS), 111, 112, 229; Lync, 36; Office 365, 35; Office Communicator, 409; OneDrive (formerly SkyDrive), 16; Outlook, 99; Sharepoint, 256
- Middleware, 33
- Midomi, 342, 379
- MiFi wireless router, 62
- Millennium ILS, 399
- Millennials, 98, 390, 413
- MIME. *See* Multipurpose Internet Mail Extensions
- MMC. *See* Malicious Mobile Code
- MMORPG. *See* Massive, Multiplayer, Online Role-Playing Games
- Mnemonic symbols, 158
- Mobile: 2.0, 145–46; accessibility, and, 153; apps, 15, 29, 102, 145, 228, 231, 378, 388, 401, 406, 411, 412; BYOD, and, 32; data services, 64–65; Internet connections, and, 44, 50, 61, 62, 64–65, 105; context, 146; design, and, 135, 142–43, 145–47, 151, 153, 156, 163, 167, 187, 217; devices, 14–19, 28, 30, 35–38, 66, 252; generations, 62; layers, 63; markup languages, 197–200; phones, 4, 7, 62; resolutions, 150, 159; screen sizes, 149, 159; security, and, 109–11, 122, 129; users and uses, 28, 109, 311, 314, 315, 378, 388, 394–95, 412–14; Web, 27–28, 38, 62, 255, 387, 408, 418
- Modems: 56K, 56; cable, 46; defined, 12, 56; DSL, 47; handshake, 56
- MODS. *See* Metadata Object Description Schema
- Modulation, signal, 11
- Monitor resolution, 163
- Monolithic systems, 269–70
- MOOC. *See* Massive Open Online Course
- Moore's Law, 14
- MooTools, 243
- Morris worm, 116
- MOSAIC, 6, 25, 203, 229, 352
- Motion Picture Experts Group (MPEG), 316–19
- MPEG. *See* Motion Picture Experts Group
- MUD. *See* Multi User Dungeon/Dimension
- Multipurpose Internet Mail Extensions (MIME), 97, 102, 320
- Multi User Dungeon/Dimension (MUD), 393
- Multiuser Virtual Environment (MUVE), 393
- MUVE. *See* Multiuser Virtual Environment
- My Cloud, 30
- MySpace, 27, 29, 405–7
- MySQL, 17, 31, 144, 230, 233, 243, 257, 258, 260, 262–65
- NaaS. *See* Networks as a service
- Namespace prefixes, 286–87
- Namespaces, 292–93
- Napster, 375–76
- NAT. *See* Network Address Translation Protocol
- National Center for Supercomputing Applications (NCSA), 6, 25, 229, 352
- National Research and Education Network (NREN), 6
- National Science Foundation (NSF), 5, 99
- National Science Foundation Network (NSFNET), 5–6, 14, 30, 56, 58, 88
- Natural language processing (NLP), 331
- Natural language search, 331, 365
- Navigational aids, 253–54
- NCSA. *See* National Center for Supercomputing Applications
- Neighbor advertisement messages, 82, 83
- Neighbor Discovery Protocol (NDP), 74, 83
- Neighbor solicitation messages, 82, 83

- Nelson, Ted, 24
- NETBIOS. *See* Network Basic Input/Output System
- Netflix, 19, 36, 38–38, 317
- Netiquette, 98, 360
- Net neutrality, 8–9, 65–67
- Netscape, 6, 25, 105, 106, 169, 176, 204, 205, 233, 237–38, 391
- Netstat command, 92, 127–29
- Network Address Translation Protocol (NAT), 54, 77, 79–80, 124
- Network Basic Input/Output System (NETBIOS), 112, 126, 129
- Network Information Centers (NIC), 8, 88
- Network Interface Card (NIC), 46, 428
- Network operating system (NOS), 52
- Networks as a service (NaaS), 34
- Newspapers, 17, 38, 166, 343, 380
- Next Generation IP (IPng), 80
- NIC. *See* Network Information Centers or Network Interface Card
- Nielsen, Jakob, 144, 155
- Nimda, 116
- NLP. *See* Natural language processing
- Nodes: in Drupal, 264–65; in HITS, 363; in networks, 5, 31, 33, 45–46, 48–49, 51–52, 60, 70; in XML, 282, RDF, and, 298, 371; social media, and, 377; topologies, and, 53–54, 60; TCP/IP, and, 70, 74, 77, 79, 83–84, 107
- NodeXL, 377
- Normalization, 258–59
- NOS. *See* Network operating system
- Novatel Wireless, 62
- NREN. *See* National Research and Education Network
- NSF. *See* National Science Foundation
- NSFNET. *See* National Science Foundation Network
- Null value, 228
- Nibble, 54
- Nyquist, Harry, 12
- Nyquist-Shannon Sampling Theorem, 12, 156, 315
- OAIS reference model, 267–69, 271, 303, 396
- Object-oriented databases, 357
- Object-oriented programming (OOP), 230
- OCLC: catalogs, and, 99, 332, 390, 398, 412; ContentDM, 270; Dublin Core, 298, 351; Ezproxy, 55; Web data, and, 349, 397
- OCR. *See* Optical character recognition
- Octal values, 432–33
- Octet, 77–78, 84, 86, 426
- ODBC. *See* Open Database Connectivity Model
- OneDrive (formerly SkyDrive), 16, 33, 35
- Online Public Access Catalog (OPAC), 99, 139, 389. *See also* Catalog
- Onlive desktop, 32, 35
- OOP. *See* Object-oriented programming
- OPAC. *See* Online Public Access Catalog
- OPALS ILS, 399
- Open Database Connectivity Model (ODBC), 343
- Open source software, 5, 16–17, 63, 112, 126, 136, 144, 230, 243, 257, 258, 260, 262–67, 270, 317, 377, 388, 399, 401–2, 409
- OpenSSL, 111–12, 114
- Open System Interconnect (OSI), 44–47, 51, 52, 63, 66, 91–92, 106
- OpenURL, 402
- Operating system (OS), 10, 14, 25; cloud, and, 33, 36, 40; CMSs, and, 262; design, and, 135, 137, 159, 182, 431; files, and, 309, 318; security, and, 110–11, 115, 117, 122, 125, 128, 129; mobile, and, 62–63; networks, and, 44, 46, 48, 51; programming, and, 229, 231; TCP/IP, and, 81, 91–92; VMs, and, 30–33
- Optical character recognition (OCR), 310, 340, 342
- O'Reilly, Tim, 7, 27, 391, 416, 418
- OS. *See* Operating system
- OSI. *See* Open System Interconnect
- Output resolution, 159, 163
- OverDrive, 138, 401, 412
- OWL. *See* Web Ontology Language
- Oysterbooks, 36, 39
- P2P. *See* Peer-to-peer
- Packet: addressing, 45–46, 53, 71–73; defined, 3, 12, 45–46, 52; encapsulation, 46, 72; forwarding, 46–47; in wireless, 50–51; net neutrality, and, 8, 65–66, IP, and, 60–71; TCP, and, 74–76, 83–86; UDP, and, 75–76
- Packet switching, 5, 46, 51–52, 69–71, 93

- Page, Lawrence, 362, 364
 PageRank, 339, 362–64, 366–70
 Password: cracking, 123; manager, 126;
 strong, 114, 123, 126; weak, 126;
 Wi-Fi, and, 51, 122; security,
 125–27; 129; theft, 109, 112,
 116, 121; tips, 126, 128
 PC. *See* Personal Computer
 PCIe. *See* Peripheral Component Interconnect Express architecture
 PCM. *See* Pulse code modulation
 PDF. *See* Portable document format
 Peer-to-peer (P2P): file sharing, 66, 396;
 networks, 45, 48; searching, and,
 375–76
 Peripheral Component Interconnect Express architecture (PCIe), 12
 Personal Computer (PC): 7, 9; architectures, 12, 13, 44; security, and, 110–11; trends, 13–15, 28, 33, 43, 62, 66
 Personalization, 358, 377–78, 381
 Petabyte, 16
 Pharming, 110, 120
 Phase modulation, 11
 Phishing, 110, 113, 116–20
 PHP: arrays, 236, 244, 246–47; examples, 245–50; global variables, 250–52; history, 243; HTML, and, 244; MySQL, and, 144, 230–31; register_globals, 114, 251–252; testing, 233; variables, 244
 Phrasing content, 200, 206
 Physical markup, 179
 Pica, 154
 PID. *See* Process identifier
 Ping, 76, 87, 92, 128
 Pinterest, 29, 30, 114, 405–08
 Pixels: defined, 159; dot pitch, and 163; image maps, and, 192–93; in color depth, 161; in CSS, 211–12, 217–19; in image search, 342; in resolution, 149, 162; points, and 154
 Plain Old Telephone System (POTS), 55, 57
 Playbook, 318
 PNG. *See* Portable Network Graphic
 Podcasts, 17, 39, 99, 100, 266, 307, 313, 314, 340, 388, 389, 409
 Point, font, 154
 Point-to-Point Protocol (PPP), 81, 91, 389
 Point-to-Point Tunneling Protocol (PPTP), 54
 Poll everywhere, 36
 Polysemy, 332–33, 337, 353
 Pony botnet, 121, 126
 POP. *See* Post Office Protocol
 Ports: communication, 13, 45; HTTP, and, 103; physical, 45, 48, 77; security, and, 109, 112–13; TCP/UDP, 74–76, 87, 92
 Portable document format (PDF): 27, 310–12; as XML output, 269, 296; eBooks, and, 311–13; search, and, 355, 359; security, and, 115
 Portable Network Graphic (PNG), 155, 164, 310
 Portal, 27, 141–42, 354, 355, 378
 Post Office Protocol (POP), 96–97, 107
 PostScript, 311–12
 POTS. *See* Plain Old Telephone System
 PPP. *See* Point-to-Point Protocol
 PPP-over-Ethernet (PPoE), 81–83
 PPTP. *See* Point-to-Point Tunneling Protocol
 Precision, 331–35
 Pref cookie, 378
 Prezi, 33, 35
 Privacy: cookies, and, 105–6, 378; search engines, and, 357, 377–78, 382; SNS, and, 119; Web 2.0, and, 415–18
 Private IP addresses, 55, 79
 Probabilistic Model, 335, 339, 362
 Procedural markup, 174, 203
 Process identifier (PID), 128
 Programming concepts, 228–29
 Proprietary tags, 176, 199, 204
 ProQuest, 379
 Protocol, 45, 51–52
 Proxy server, 54–55, 79, 87, 93, 102, 112, 124
 PSTN. *See* Public Switched Telephone Network
 Public relations in Web design, 138–39
 Public Switched Telephone Network (PSTN), 57
 Pulse code modulation (PCM), 11
 QBE. *See* Query by example
 QBIC. *See* Query by image content
 QR codes, 413

- Query by example (QBE), 259, 342
 Query by image content (QBIC), 342
 Query processor, 361–62
- RARP. *See* Reverse Address Resolution Protocol
 Raster images, 160–61
 RBSE. *See* Repository-based software engineering
 RDA. *See* Resource Description and Access
 RDF Site Summary (also Really Simple Syndication or Rich Site Summary). *See* Really Simple Syndication
 RDF triple, 298
 RDF. *See* Resource Description Framework
 RDF. *See* Resource Description Framework
 RDFA. *See* Resource Description Framework in Attributes
 Really Simple Syndication (RSS), 17, 27, 29, 98, 107, 144, 275, 297, 300, 313, 315, 377, 388, 409, 411–12
 Real-Time Streaming Protocol (RTSP), 100, 317
 Real-Time Transport Protocol (RTP), 99–100, 317
 Recall, 334–35
 Rehabilitation Act Amendments of 1998 Section 508, 166
 Relational databases, 230, 256–59, 260
 Relative reference, 183–87, 201
 Repository-based software engineering (RBSE), 353
 Representational State Transfer (REST), 27
 Resolution issues, 162–63
 Resource Description and Access (RDA), 395–96
 Resource Description Framework (RDF), 276–77, 297–99, 303, 371, 389, 395
 Resource Description Framework in Attributes (RDFA), 277, 298–99, 303, 351, 371, 374, 389, 396
 Resource evaluation, 320–23
 Responsive design, 146, 149, 156, 169, 204, 212, 214–19, 225
 REST. *See* Representational State Transfer
 Reverse Address Resolution Protocol (RARP), 81–82
 RGB color, 161–62, 213
 Rich Site Summary (also RDF Site Summary or Really Simple Syndication). *See* Really Simple Syndication
 Ring networks, 48, 53
 Root directory, 184–85
 Root element, 177, 178, 181, 197, 282, 284–89, 292
 Rootkit, 115, 117, 123, 128
 Routers: ARP, and, 82, 53; IP, and, 76, 80, 81; NAT, and, 79; network, 45–47, 59; proxy servers, and, 87; security, and, 112, 122, 124; SNMP, and, 107; subnet masks, and, 83–87; Wi-Fi, and, 61, 62
 RSS. *See* Really Simple Syndication
 RTP. *See* Real Time Transport Protocol
 RTSP. *See* Real Time Streaming Protocol
 Ruby on Rails, 230, 256
 Rule of Cs, 390
 Russian Doll Model, 287–90, 302
 Rustock botnet, 96
 SaaS. *See* Software as a service
 Same origin policy, 113, 301
 Sampling technologies, 12, 156, 315–16
 SAN. *See* Storage Area Network
 Sans serif fonts, 155
 Satellite Internet service, 60
 Scalable Vector Graphic (SVG), 155–56, 164, 165, 275, 297, 310
 Scarcity problem, 363
 Schema: database, 175; Definition Language, 283, 286–87; examples, 287–90, 297; in XML, 175, 277; namespaces, and, 293
 Schematic, in design, 136, 147–48, 159, 169
 Script programming: arrays, 236, 237, 244, 246–47; concatenation, and, 236, 240; defined, 231–33; event handlers, 237; functions, 240–41; languages, 228–30; parameters, 231, 232; subroutine, 228; variables, 228–29, 236
 Search engine optimization (SEO), 168, 170, 350, 360, 369, 372
 Search engine results page (SERP), 168, 358
 Search engines: anatomy, 359–63; challenges, 348–51; economics, 356–57; history, 351–53; indexical bias, 358; index creation, 360–61; inverted files, and, 360–61; types, 353–56

- Second Life (SL), 389, 393–94
 Secure Sockets Layer (SSL), 95, 106, 112, 114
 Segments, 45–47, 83
 Semantic search: 276–77, 355; Bing, and, 372, 375; Google, and, 272, 373–74, 367; personalization, and, 377; technologies for, 297, 298, 328, 351, 365, 366, 370–72
 Semantic Web, 276, 297, 298, 328, 332, 370–71, 381, 391, 393, 395–96, 401
 SEO. *See* Search engine optimization
 Serial Line Internet Protocol/Point-to-Point Protocol (SLIP/PPP), 81, 389
 Serif fonts, 155
 SERP. *See* Search engine results page
 Server Side Includes (SSI), 229, 234, 255, 261, 312, 431
 Server-side processing: design, and, 136–37; image maps, and, 158, 193; security, and, 114; web programming, and, 104, 188, 233–35, 243, 258, 431
 Service Set Identifier (SSID), 51
 Servlet: container, 230; Java, and, 230, 234
 Session Initialization Protocol (SIP), 100
 SETI@home, 33, 276
 SFX, 398, 402
 SGML. *See* Standard Generalized Markup Language
 Shannon, Claude, 12
 Shared bookmarking, 376, 407, 411–12
 Shazam, 342, 379
 Shell account, 25, 55, 66, 81, 96, 100
 Shock sites, 141
 Simple Internet Protocol Plus (SIPP), 80
 Simple Mail Transfer Protocol (SMTP), 95–97, 107
 Simple Network Management Protocol (SNMP), 51
 Simple Object Access Protocol (SOAP), 27, 229, 275, 297
 SIP. *See* Session Initialization Protocol
 SIP. *See* Session Initialization Protocol or also Submission Information Package
 SIPP. *See* Simple Internet Protocol Plus
 Siri, 331, 365, 372, 382
 Skype, 36, 100
 SL. *See* Second Life
 SLIP/PPP. *See* Serial Line Internet Protocol/Point-to-Point Protocol
 Smart agents, 377–78, 393
 Smartphones: apps, and, 231, 396; history, 7, 62; resolutions, 149; sales, 15, 62; screen sizes, 150. *See also* mobile devices
 SMTP. *See* Simple Mail Transfer Protocol
 Snapchat, 30, 35, 405
 SNMP. *See* Simple Network Management Protocol
 Snopes, 128
 SNS. *See* Social Networking Services
 SOAP. *See* Simple Object Access Protocol
 Sobig worm, 116
 Social capital, 28, 404–6
 Social engineering, 110, 118, 126
 Social media: 7, 15, 16, 28–30, 35; business, and, 38; design, and 138, 150, 159, 262; opportunities, 390, 400, 406–11; privacy, and, 416–17; search, and, 366–67, 382; security, and, 114, 118, 127
 Social Networking Services (SNS): activities, 98, 314–315, 321, 406–11; sites, 29–30, 405–7
 Social signal, 366, 370, 372–73, 383
 Social web, 28, 30, 40, 396, 405, 408, 409
 Software as a service (SaaS), 34, 395, 399, 414
 Spam: dangers, 113, 116–120, 122; defined, 96; filtering, 120; statistics, 19, 96, 121
 Spambot, 110, 116, 117, 121
 Spider program, 353, 359, 360
 Splash page, 157
 Sponsored links, 141, 357, 361
 Spoofing: email, 110, 116, 118; program messages, 117; Wi-Fi, 109, 121–22; URLs, 113, 119
 Sprint, 8, 56, 64
 Spyware, 110, 113, 121, 124
 SQL. *See* Structured Query Language
 SSI. *See* Server Side Includes
 SSID. *See* Service Set Identifier
 SSL. *See* Secure Sockets Layer
 Standard Generalized Markup Language (SGML), 26, 174–75, 177, 199, 277, 278, 283, 302
 Star topology, 46, 48–49
 Statelessness, 105–7
 Sticky sites, 141

- Stopwords, 330, 332
 Storage Area Network (SAN), 31
 Storyboarding, 147
 Streaming: bandwidth, and, 14, 66, 317; net neutrality, and, 8; providers, 35, 38, 389, 406–7; technologies, 29, 75–76, 99–100
 Strict DTD, 178–79, 299
 Structured Query Language (SQL), 230–31, 257, 259, 265, 327
 Style languages, 203–4, 208, 294
 Style tiles, 149
 Submission Information Package (SIP), 268–69
 Subnet: administration, 83–84; mask, 13, 47, 69, 82–87
 Surrogate record, 325, 327–28
 SurveyMonkey, 36, 414
 SVG. *See* Scalable Vector Graphic
 Swen worm, 116–17
 SWF. *See* Flash file format
 Swiss Army knife Web design, 142
 Switch (network), 47, 54, 81, 83
 System layers, 44, 46, 91
 T1 lines, 14, 55, 58 59, 390
 T3 lines, 57–58
 Tablet. *See* Mobile device
 Tag cloud, 331, 340–42
 Tagging, 340, 389, 400, 403–4, 411
 Task tagging, 411
 TCP. *See* Transmission Control Protocol
 TCP/IP. *See* Transmission Control Protocol/Internet Protocol
 Techshops, 404
 TechSmith Camtasia Studio, 318, 414
 Telnet, 13, 74, 95, 98–99
 TeraGrid, 33
 Term: leasing, 357; selection, 331–32; stemming, 337, 381; stuffing, 168, 350; weighting, 154, 278, 332, 337, 339, 361
 Text decoration, 154, 178, 278, 366, 367
 Text Retrieval Conference (TREC), 327
 The Information Mine (TIM), 23
 Thesaurus, 331, 333–34, 337, 342, 353, 379
 Thin clients, 30, 32
 3D printers, 418
 Tier 1 network, 56
 Tier 2 network, 56
 Tier 3 network, 56
 TIFF. *See* Aldus Tagged Image
 TIM. *See* The Information Mine
 Time to Live (TTL), 72
 Time Warner, 60
 Tineye, 379
 TinyMCE, 260
 TinyURL, 26, 141, 402
 TLD. *See* Top-Level Domain
 TN3270, 99
 Token ring, 48, 53
 Top-level domain (TLD), 6, 53–54
 Topology, 46–49
 Traceroute (also tracert) command, 76, 92
 Transitional DTD, 175, 178, 179, 299
 Transmission Control Protocol (TCP): defined, 70–71, ports, 74–76, 95–96, 61; SYN-ACK three-way handshake, 74, 113
 Transmission Control Protocol/Internet Protocol (TCP/IP): command utilities, 92; defined, 69–70; history, 5; layers, 44, 91–92
 TREC. *See* Text Retrieval Conference
 Tree: directory, 184–85; network, 48; nodes in XML, 282
 Trillian, 408
 Trojan Horse, 97, 110, 115–17, 122, 123, 127, 320
 TTL. *See* Time to Live
 Tumblr, 30, 405–7
 Turnkey systems, 399
 Tweets, 28, 29, 313, 314, 323
 Twitter, 17, 18, 28, 29–30, 121, 300, 314, 321, 366, 372, 396, 405–9, 416
 Typography, 154–55, 212
 UART. *See* Universal Asynchronous Receiver Transmitter
 UC3. *See* California Digital Library Curation Center
 UDP. *See* User Datagram Protocol
 UGC. *See* User generated content
 UMTS. *See* Universal mobile telecom system
 Unicode, 277, 308, 328
 Uniform Resource Identifier (URI), 24, 25, 292, 298, 371
 Uniform Resource Locator (URL), 24–26, 90–91, 140–41
 Uniform Resource Name (URN), 292
 Universal Asynchronous Receiver Transmitter (UART), 56–57
 Universal Mobile Telecom System, 64

- Universal Serial Bus (USB): 45, 47, 159; modem, 62
- UNIX: issues, 186–87; permissions, 100, 432–33; servers, 431; superuser, 114
- Unshielded twisted pair (UTP), 49, 53
- URI. *See* Uniform Resource Identifier
- URL. *See* Uniform Resource Locator
- URN. *See* Uniform Resource Name
- Usability: catalogs, and, 400; design, and, 138–40, 143–45, 155, 168; search, and, 355; testing, 136
- USB. *See* Universal Serial Bus
- Usenet, 28, 98, 313, 314
- User 2.0, 400, 404
- User Datagram Protocol (UDP), 70, 72, 74–76
- User experience design (UXD), 143
- User generated content (UGC), 28, 404
- User interface: design of, 143–45; search engines, and, 355–56, 359
- UTP. *See* Unshielded twisted pair
- UUENCODE, 97, 320
- U-verse, 60
- UXD. *See* User experience design
- Validity, XML, 277, 282–83
- Vanity publishing, 321
- VDI. *See* Virtual desktop infrastructure
- Vector graphics, 155–56, 159–61, 165
- Vector Space Retrieval Model, 335, 337–39, 362
- Verizon, 8, 9, 56, 60
- VERONICA, 101, 352, 381
- Vertical resolution, 159
- Video: conferencing, 36, 409; formats, 319; sharing, 408–11, 413–14; streaming, 8, 18, 66, 75, 100, 317; providers, 38, 317; search, and, 339, 342, 375, 377, 379
- Viewport, 212, 216–17
- Vimeo, 36, 414
- Viola browser, 203
- Virtual desktop infrastructure (VDI), 32
- Virtualization, 30–32, 34, 38, 40
- Virtual machine (VM), 30–32
- Virtual Private Network (VPN), 34, 54, 87, 122, 124
- Virtual reality (VR), 393–94
- Virus, 97, 110, 111, 115–16, 120, 123
- Visual Basic, 113, 115, 231–33, 312
- Visualization: folksonomies, and, 341, 403; search results, and, 305, 376–77, 381
- VM. *See* virtual machine
- VMware, 32
- Voice Over IP (VoIP), 8, 14, 99–100
- Voice search, 331, 365, 372, 375, 382
- VoIP. *See* Voice Over IP
- Voyager ILS, 399
- VPN. *See* Virtual Private Network
- VR. *See* Virtual reality
- W3C. *See* World Wide Web Consortium
- WAI. *See* Web Accessibility Initiative
- WAIS. *See* Wide area information server
- WAMP, 233, 262
- WAN. *See* Wide-Area Network
- WAP. *See* Wireless Access Protocol
- WAP2. *See* Wireless Access Protocol 2
- Watering hole attack, 123
- Wave amplitude, 10–11, 315
- Wave frequency, 10–11
- Wayback machine, 349
- W-CDMA. *See* Wideband code division multiple access
- WCM. *See* Web content management
- Web. *See* World Wide Web
- Web 1.0: content, 307; defined, 390; libraries, and, 390–91; transition of, 387, 389
- Web 2.0: content, 27, 249; critiques, of, 415–18; defined, 27, 391–92; design, and, 143–45; transition of, 387–89
- Web 3.0: descriptions, 222, 392–96; Semantic Web, and, 395–96; XML, and, 276, 278, 328
- Web Accessibility Initiative (WAI), 166–67, 201
- Web appliance, 396
- Webbrain, 377
- Web bug, 105, 120–21
- Web commerce. *See* eCommerce
- Web design: accessibility, and, 136, 137, 151, 153, 157, 166–68; branding, and, 136, 138, 141; color, and, 155, 157–58; findability, and, 168; fonts, and, 154–55; images, and, 158–61; mobile, and, 146–47; myths, of, 142; navigation, and, 140–41, 153–54; needs analysis, for, 138; performance-based, 156; responsive, and, 146; Rule of Seven, and, 142; schematics, for, 146–48; Three Click Rule, and, 147, 151; Web 2.0, and, 143–45

- Web Hypertext Application Technology Working Group (WHATWG), 199
- Web Ontology Language (OWL), 276, 395
- Web programming: databases, and, 192–195; history, 191–92; JavaScript, and, 202–3; languages, 191, 192, 196; PHP, and, 208–9
- Webspm, 370
- Weebly, 262, 263
- WEP. *See* Wired Equivalent Privacy
- WHATWG. *See* Web Hypertext Application Technology Working Group
- Whois command, 88
- Wide area information server (WAIS), 352
- Wide-Area Network (WAN), 44, 399
- Wideband code division multiple access (W-CDMA), 64
- Widgets, 113, 201, 242, 252, 388, 391, 413
- Wi-Fi. *See* Wireless fidelity
- Wi-Fi Protected Access (WPA), 61
- Wi-Fi Protected Access 2 (WPA2), 61, 122
- Wiki, 17, 27, 29, 323, 389, 392, 409–10
- Wikipedia, 18, 29, 39, 266, 315, 374, 406, 410
- WiMax, 64–65
- Winsock, 44, 81
- Wintel: defined, 9–10; design, and, 169; security, and, 109
- Wired Equivalent Privacy (WEP), 61, 122
- Wireframe, 136, 147, 149, 151
- Wireless Access Protocol (WAP), 61, 198
- Wireless Access Protocol 2 (WAP2), 198
- Wireless fidelity (Wi-Fi), broadband, 61–62, libraries, and, 66 standards, 50, 65; security, 109, 121–22, 129
- Wireless LAN (WLAN), 50–51
- Wireless Markup Language (WML), 197–98, 201
- WIX, 262, 263
- WLAN. *See* Wireless LAN
- WML. *See* Wireless Markup Language
- WolframAlpha, 372
- WordPress, 29, 144, 262, 263
- World Wide Computer, 388, 394–95
- World Wide Web (WWW): defined, 23–25; cloud computing, and, 33–36; future, of, 387–93; history, 6–7, 23–25; size of, 348–49; statelessness, of, 105–6; technologies, 25–26; use of, 37–40
- World Wide Web Consortium (W3C): Box Model, 215–16; CSS, and, 204; described, 7; DOM standard, 238; HTML, and, 176–78; HTML5, and, 199; OWL, and, 276, 395; Semantic Web, and, 276; Web Accessibility Initiative, and, 166–67; Web administration, and, 8; XML, and, 276–77; XSDL, and, 286; XSL, and, 296
- Worm, 47, 97, 110, 112, 115–17, 123, 127, 320, 354
- WPA. *See* Wi-Fi Protected Access
- WPA2. *See* Wi-Fi Protected Access 2
- WWW. *See* World Wide Web
- Xanadu, 24
- Xbox, 372
- XenDesktop, 32
- XHTML standard, 196–97, 297
- XHTML-MP, 198
- XML (Extensible Markup Language): alternatives, 300; CSS, and, 294–95; data structuring, and, 280–91; declarations in, 282; described, 275–80; DTD, and, 283–86; graphics, and, 165; HTML, comparison with, 277–78; elements in, 284–85; JSON, comparison with, 300–302; namespaces, 286–87, 292–94; namespace prefixes, 286; RDF, and, 298–99; RDFa, and, 298–99; related technologies, 296–97; Schema Definition, and, 286–87; specification, 279–80; validity, and, 282–83; viewing of, 293–95; well-formedness, and, 282–83; XHTML, and, 299–300
- XML Schema Definition (XSD), 280, 286–87
- XML Schema Description Language (XSDL), 283, 286–87
- XSD. *See* XML Schema Definitions
- XSDL. *See* XML Schema Description Language
- XSL (Extensible Stylesheet Language), 296, 302
- XSLT (XSL Transformations), 198, 279, 280, 296
- Yahoo: Bing, and, 375; directory, 353–54; Flickr, and, 36, 410; IM, 408; mail, 97; market share, 357, 364; mashups, and, 411; portal,

496 Index

- 378; search, 358, 371, 372; tools, 88, 243; Tumblr, and, 407
- YAML, 276, 300, 303
- Yottabyte, 16
- YouTube, 36, 39, 144, 165, 315, 318, 340, 342, 364, 377, 379, 389, 408, 411, 414
- Zero day vulnerability, 111
- Zillow, 412
- Zinio, 39, 138, 401
- Zipf's Law, 372
- Zipf's Principle of Least Effort, 335
- Zoho, 33, 35
- Zombie, 113, 116, 121

About the Author

JOSEPH B. MILLER, MSLS, is associate professor emeritus at the University of Kentucky School of Library and Information Science, College of Communication and Information, Lexington, KY, where he devoted the majority of his efforts to teaching LIS information technology courses and exploring instructional technologies to advance the school's online degree program. His published works include the first edition of Libraries Unlimited's *Internet Technologies and Information Services* and the articles "The Internet Has Changed like, *Everything*" and "PC Security in a Networked World." He is coauthor (with Donald Case) of "Do Bibliometrists Cite Differently from Other Scholars." He is a member of the Beta Phi Mu Honorary Society, recipient of the College of Communications and Information Studies Excellence in Teaching Award (2000), and a past president of the Kentucky chapter of the Special Libraries Association. Miller received his master of science in library science degree from the University of Kentucky.