# Index

<table>
<thead>
<tr>
<th>A</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>469</td>
</tr>
<tr>
<td>Abstract Factory Method</td>
<td>34, 373</td>
</tr>
<tr>
<td>Abstractions</td>
<td>23, 24, 255</td>
</tr>
<tr>
<td>Abstract windowing toolkit (AWT)</td>
<td>251</td>
</tr>
<tr>
<td>ActionListener interface</td>
<td>91</td>
</tr>
<tr>
<td>actionPerformed() method</td>
<td>114, 144</td>
</tr>
<tr>
<td>ActiveX</td>
<td>30, 33</td>
</tr>
<tr>
<td>act() method</td>
<td>174, 415</td>
</tr>
<tr>
<td>Actor</td>
<td>12, 322–324, 335–336</td>
</tr>
<tr>
<td>Adapter</td>
<td>34</td>
</tr>
<tr>
<td>addScore() method</td>
<td>300–302</td>
</tr>
<tr>
<td>Advanced mobile phone system (AMPS)</td>
<td>258</td>
</tr>
<tr>
<td>Advertisements</td>
<td>451</td>
</tr>
<tr>
<td>Aggregate entity (AE) pattern</td>
<td>228</td>
</tr>
<tr>
<td>enhanced</td>
<td>229</td>
</tr>
<tr>
<td>Aliases</td>
<td>417</td>
</tr>
<tr>
<td>Analog wireless phones</td>
<td>260</td>
</tr>
<tr>
<td>Applets</td>
<td>289</td>
</tr>
<tr>
<td>application identifier</td>
<td>289</td>
</tr>
<tr>
<td>DigitalMoney</td>
<td>289</td>
</tr>
<tr>
<td>Java see Java applet</td>
<td>289, 291</td>
</tr>
<tr>
<td>Java Card</td>
<td>290</td>
</tr>
<tr>
<td>lifecycle of</td>
<td>290</td>
</tr>
<tr>
<td>VirtualCurrency. See VirtualCurrency applet</td>
<td>10</td>
</tr>
<tr>
<td>Application Assembler</td>
<td>10</td>
</tr>
<tr>
<td>Application Component Provider</td>
<td>10</td>
</tr>
<tr>
<td>Application identifier (AID)</td>
<td>289</td>
</tr>
<tr>
<td>Application programmers</td>
<td>390</td>
</tr>
<tr>
<td>Application programming interface (API)</td>
<td>232</td>
</tr>
<tr>
<td>communication protocols</td>
<td>306</td>
</tr>
<tr>
<td>connected limited device configuration</td>
<td>426–432</td>
</tr>
<tr>
<td>Cyc</td>
<td>53</td>
</tr>
<tr>
<td>description of</td>
<td>287</td>
</tr>
<tr>
<td>Java Card</td>
<td>160</td>
</tr>
<tr>
<td>Java class implementations</td>
<td></td>
</tr>
<tr>
<td>Java Speech</td>
<td>133</td>
</tr>
<tr>
<td>mobile media</td>
<td>309</td>
</tr>
<tr>
<td>proprietary types</td>
<td>60–62</td>
</tr>
<tr>
<td>record management system</td>
<td>310</td>
</tr>
<tr>
<td>stability of</td>
<td>232</td>
</tr>
<tr>
<td>wireless messaging</td>
<td>312</td>
</tr>
<tr>
<td>Application scenario language (ASL)</td>
<td>417</td>
</tr>
<tr>
<td>aliases</td>
<td>417</td>
</tr>
<tr>
<td>condition element</td>
<td>418–420</td>
</tr>
<tr>
<td>definition of</td>
<td>416</td>
</tr>
<tr>
<td>parent-umbrella for objects</td>
<td>422</td>
</tr>
<tr>
<td>prompting user for an answer</td>
<td>416</td>
</tr>
<tr>
<td>query element</td>
<td>420</td>
</tr>
<tr>
<td>translate element</td>
<td>418</td>
</tr>
<tr>
<td>Architects</td>
<td>9–10</td>
</tr>
<tr>
<td>Architectural views</td>
<td>46</td>
</tr>
<tr>
<td>Architecture</td>
<td>44</td>
</tr>
<tr>
<td>asynchronous</td>
<td>44</td>
</tr>
<tr>
<td>client-server</td>
<td>28, 43</td>
</tr>
<tr>
<td>construction of</td>
<td>28</td>
</tr>
<tr>
<td>contract-based</td>
<td>43</td>
</tr>
<tr>
<td>data-centered</td>
<td>46</td>
</tr>
<tr>
<td>data-flow</td>
<td>44</td>
</tr>
<tr>
<td>data grid</td>
<td>46</td>
</tr>
<tr>
<td>distributed, decentralized</td>
<td>46</td>
</tr>
<tr>
<td>distributed active knowledge base</td>
<td>46</td>
</tr>
<tr>
<td>event-driven</td>
<td>45</td>
</tr>
<tr>
<td>knowledge driven</td>
<td>46</td>
</tr>
<tr>
<td>layered client-server</td>
<td>43</td>
</tr>
<tr>
<td>Model-View-Controller</td>
<td>34, 43, 223</td>
</tr>
<tr>
<td>multi-tier</td>
<td>28, 30–31, 39</td>
</tr>
<tr>
<td>procedure-based</td>
<td>44</td>
</tr>
<tr>
<td>remote method invocation</td>
<td>57</td>
</tr>
<tr>
<td>representational state transfer</td>
<td>43</td>
</tr>
<tr>
<td>service-based</td>
<td>58</td>
</tr>
<tr>
<td>service-oriented</td>
<td>44–45</td>
</tr>
<tr>
<td>single-user</td>
<td>28</td>
</tr>
<tr>
<td>Index</td>
<td>Page</td>
</tr>
<tr>
<td>-------</td>
<td>------</td>
</tr>
<tr>
<td>C++ reflection</td>
<td>510</td>
</tr>
<tr>
<td>Capability Maturity Model</td>
<td>4</td>
</tr>
<tr>
<td>goals of</td>
<td>4</td>
</tr>
<tr>
<td>levels of maturity recognized by</td>
<td>4</td>
</tr>
<tr>
<td>overview of</td>
<td>4</td>
</tr>
<tr>
<td>Card acceptance device (CAD)</td>
<td>288</td>
</tr>
<tr>
<td>Carrier sense multiple access with collision avoidance (CSMA/CA)</td>
<td>261</td>
</tr>
<tr>
<td>Case</td>
<td>471</td>
</tr>
<tr>
<td>Catch</td>
<td>471</td>
</tr>
<tr>
<td>Cell broadcast service (CBS)</td>
<td>312</td>
</tr>
<tr>
<td>Chain of responsibility design pattern</td>
<td>235</td>
</tr>
<tr>
<td>Char</td>
<td>472</td>
</tr>
<tr>
<td>aspect() method</td>
<td>235</td>
</tr>
<tr>
<td>checkPIN() method</td>
<td>295</td>
</tr>
<tr>
<td>Circuit-switched networks</td>
<td>260</td>
</tr>
<tr>
<td>Classes</td>
<td>16, 23</td>
</tr>
<tr>
<td>adding of</td>
<td>246</td>
</tr>
<tr>
<td>WebNewsContainer</td>
<td>109</td>
</tr>
<tr>
<td>Classes/responsibilities/collaborators (CRC) approach</td>
<td>15</td>
</tr>
<tr>
<td>Classification and regression trees (CARTs)</td>
<td>138</td>
</tr>
<tr>
<td>Client administration</td>
<td>255</td>
</tr>
<tr>
<td>classes added to</td>
<td>246</td>
</tr>
<tr>
<td>features added to</td>
<td>246</td>
</tr>
<tr>
<td>functionality of</td>
<td>224</td>
</tr>
<tr>
<td>independence from server</td>
<td>247</td>
</tr>
<tr>
<td>“peers”</td>
<td>248, 255</td>
</tr>
<tr>
<td>services for</td>
<td>249</td>
</tr>
<tr>
<td>thin. See Thin clients types of</td>
<td>250–251</td>
</tr>
<tr>
<td>Client devices</td>
<td>223</td>
</tr>
<tr>
<td>Client-server assumptions regarding</td>
<td>222</td>
</tr>
<tr>
<td>development of</td>
<td>222</td>
</tr>
<tr>
<td>interactions</td>
<td>224</td>
</tr>
<tr>
<td>value object assembler pattern added to</td>
<td>228</td>
</tr>
<tr>
<td>Client-server architecture</td>
<td>28, 43</td>
</tr>
<tr>
<td>Cluster unit selection</td>
<td>140</td>
</tr>
<tr>
<td>CMULexicon</td>
<td>139</td>
</tr>
<tr>
<td>CMUVoice</td>
<td>136</td>
</tr>
<tr>
<td>“Codat”</td>
<td>449</td>
</tr>
<tr>
<td>Code division multiple access (CDMA)</td>
<td>239–260</td>
</tr>
<tr>
<td>Collaborative engineering</td>
<td>1–2</td>
</tr>
<tr>
<td>Collaborative services</td>
<td>351</td>
</tr>
<tr>
<td>Collaborative systems</td>
<td>46</td>
</tr>
<tr>
<td>Common gateway interface (CGI)</td>
<td>29</td>
</tr>
<tr>
<td>description of</td>
<td>29</td>
</tr>
<tr>
<td>drawback of</td>
<td>29</td>
</tr>
<tr>
<td>Common language runtime (CLR)</td>
<td>194, 468</td>
</tr>
<tr>
<td>Common object request broker architecture (CORBA)</td>
<td>50–52, 57–58</td>
</tr>
<tr>
<td>IILop communication protocol</td>
<td>58</td>
</tr>
<tr>
<td>Component</td>
<td>43</td>
</tr>
<tr>
<td>Component object model (COM)</td>
<td>30</td>
</tr>
<tr>
<td>software</td>
<td>42</td>
</tr>
<tr>
<td>styles</td>
<td>43–46</td>
</tr>
<tr>
<td>synchronous</td>
<td>44</td>
</tr>
<tr>
<td>system</td>
<td>28</td>
</tr>
<tr>
<td>tiered</td>
<td>43</td>
</tr>
<tr>
<td>VoiceXML</td>
<td>366</td>
</tr>
<tr>
<td>Architecture analysis</td>
<td>27–28</td>
</tr>
<tr>
<td>Architecture definition languages (ADL) distributed architecture description languages (DADL)</td>
<td>47</td>
</tr>
<tr>
<td>UniCon</td>
<td>47</td>
</tr>
<tr>
<td>Argument</td>
<td>167</td>
</tr>
<tr>
<td>Arity</td>
<td>167</td>
</tr>
<tr>
<td>Aspect</td>
<td>24</td>
</tr>
<tr>
<td>Aspect-oriented programming (AOP)</td>
<td>24</td>
</tr>
<tr>
<td>Assert</td>
<td>470</td>
</tr>
<tr>
<td>Assertions</td>
<td>170</td>
</tr>
<tr>
<td>assignEventHandlers() method</td>
<td>114</td>
</tr>
<tr>
<td>Asynchronous architecture</td>
<td>44</td>
</tr>
<tr>
<td>Asynchronous communications</td>
<td>243</td>
</tr>
<tr>
<td>Asynchronous transfer mode</td>
<td>261</td>
</tr>
<tr>
<td>AudioFactory</td>
<td>344</td>
</tr>
<tr>
<td>Audio–video interfaces AVIFactory implementations</td>
<td>342</td>
</tr>
<tr>
<td>description of</td>
<td>339</td>
</tr>
<tr>
<td>Authenticator</td>
<td>261</td>
</tr>
<tr>
<td>AVFactory class</td>
<td>395</td>
</tr>
<tr>
<td>B</td>
<td>47</td>
</tr>
<tr>
<td>Behavioral view of architecture</td>
<td>47</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>262</td>
</tr>
<tr>
<td>Boolean</td>
<td>470</td>
</tr>
<tr>
<td>Break</td>
<td>470</td>
</tr>
<tr>
<td>Browsers web. See Web browsers XML</td>
<td>249</td>
</tr>
<tr>
<td>build() method</td>
<td>82, 98, 121, 127, 142</td>
</tr>
<tr>
<td>“buildPages” module</td>
<td>279</td>
</tr>
<tr>
<td>Business experts</td>
<td>391</td>
</tr>
<tr>
<td>Business process execution language for Web services</td>
<td>544</td>
</tr>
<tr>
<td>Bytes</td>
<td>471, 576–577</td>
</tr>
<tr>
<td>C</td>
<td>22, 37, 134</td>
</tr>
<tr>
<td>base class</td>
<td>487</td>
</tr>
<tr>
<td>data retrieval from the Web using graphical user interface</td>
<td>494</td>
</tr>
<tr>
<td>Java versus</td>
<td>515</td>
</tr>
<tr>
<td>keywords</td>
<td>469–484</td>
</tr>
<tr>
<td>networking</td>
<td>491</td>
</tr>
<tr>
<td>reading and writing files in</td>
<td>491</td>
</tr>
<tr>
<td>server socket listener in</td>
<td>499</td>
</tr>
<tr>
<td>sockets in</td>
<td>504</td>
</tr>
<tr>
<td>speech recognition in</td>
<td>589</td>
</tr>
<tr>
<td>subclasses</td>
<td>487</td>
</tr>
</tbody>
</table>
Index

Composite design pattern 225
Composite view design pattern 225
condition element 418–420
Configuration file 90
damage to 113
XML 103
configure() method 111
Connected device configuration (CDC) 306
Connected limited device configuration (CLDC) 306
Connectors 43
Constants 169
Constructor 79, 325
Container 43, 56
Context-free grammar (CFG) 361
Continue 472
ContourGenerator 139
Contract-based architecture 43
ControlComponent 95, 98, 127
building of 104
constructor of 95
parameters for 105
source code for 90
VoiceComponent and 140–143
Controller 34
Control systems and controllers
description of 199–203
legacy systems 199
messaging systems 201–203
middle level 199
robots 199
subsystems 200
unified controller 201
Core module-level requirements 23
Cyc application programming interface 426–432
CycHTMLFactory 411–412, 415
CycL language
advantages of 163
argument 167
Arity 167
assertions 170
characteristics of 165
complicated logic expressed by 165
components of 165
constants 169
core theories 169
creating rules in 170
development of 163
domain-specific theories 169
Events 171
expressiveness of 164
functions 169
ground-level facts 169
knowledge base 168–169, 249
learning about 165
microtheories 170–171
Natural Language-to-Cyc 189
predicates 169
projections 164
question answering by 173
roles 171–172
status of 164
syntax of 169
unary predicates 168
upper ontology layer 169
CycML 541
Cyc responses 412
D
DAML class 154
description of 153–156
DARPA Agent Markup Language. See DAML
Darwin 47
data access object (DAO) 34
data-centered architectures 46
dataConnector abstract class 196
Data-flow architecture 44
data grid architecture 46
data integrator 198
data mining
description of 160–161
steps involved in 160
data mining engine (DME) 160
DataObject class 204
DataReference class 204, 209
data sources 196
data storage
mobile devices 310
types of 193–194
Decorator design pattern 223
Default 472
demilitarized zone
definition of 32
firewalls 32
Deployer 10
Deployment view of architecture 47
deselect() method 294
Designers 9–10
Designer design pattern
description of 34, 223
Observer 45
developers
extreme programming 5
roles of 9–10
team communication among 39
Dialog Manager 161
digital cellular technologies 260
digitalMoney applet 289
digital signal processor (DSP) 366
diphone unit selection 140
Dispatcher 34
dispatch() method 99, 114–116
Index

Segmenter 138
Tokenization 137
TokenToWords 139
UnitConcatenator 139
UnitSelector 139
utterance 133–136
UtteranceProcessors 135, 136, 137
Voice 135, 147
FreeTTSSpeakable 135
Frequency division multiple access (FDMA) 258
Functional view of architecture 47

G
General packet radio service (GPRS) 260
Geographical information systems 570–575
getAllGroupPrivileges() method 295
cGetConnection() method 196
cGetEventSources() method 114, 122–123, 129, 147
cGetGroupIndex() method 300
cGetGroupRoles() method 297
cGetIdentity() method 304
cGetRecordType() method 331
cGetTypeIndex() method 333
cGetVoices() method 339
Global system for mobile (GSM) 259–260
GNU network object model environment (GNOME) 250
Graphical user interface (GUI) 82, 251
C++ 515, 531
Java 515, 524
terms to define 533

H
Hashtable 111, 237
Hashtable "nextStepRules" 237
HashtablenextStepRules 237
HesselBeim, Frances
HyperLAN1 261
HyperLAN2 261
Hyper Text Markup Language (HTML) 226, 234, 268
Hyper Text Transfer Protocol (HTTP) 59
Hyper Text Transfer Protocol (HTTP) tunnelling 456

I
Ideal comprehensive system (ICS) 164
Identity key 288
If 473
Implements 475
Import 475
Inference engine 388
Inference frameworks 388
Information management 158
Inheritance 16–21
init() method 74, 80, 98, 235
initRecordScenario() method 331
Instanceof 473
Instant screen share 581
Integration-ready systems development 1
developers 2
management style and 2
Intelligent machines 388
Interaction view of architecture 47
Interface 476
Interface definition language (IDL) description of 51
skeltons 52
stubs 52
International mobile telecommunications-2000 260
Internet 7, 222, 247
interpret() method 200
Intonator 138
invoke() method 330
ISO 9000 4
Item 136
Java
base class 486
C++ versus 484
data retrieval from the Web using 493
graphical user interface 515, 524
keywords 469–484
networking 491
reading and writing files in 491
server socket listener in 497
sockets in 501
speech recognition with 348–349
style guidance 37
subclasses 486
Java abstract windowing toolkit (AWT) 251
Java applet 73, 70
communication with server 109
example of 516
init() method 74, 80, 98
main() method 82
Moving News 70–79
mulitselectable 287
restrictions 226
run() method 75
start() method 75, 90
stop() method 75, 80
tags 71
while (true) loop 73
writing of 72

JavaCard
applets 289, 291
application programming interface 287
card reader for 288
description of 287
identity key 288
javacard framework package 290
limitations 290
programming of 288–290
strategy for developing applications using 290
Java and framework
Java Card 2.2 Runtime Environment (JCRE) specification 288, 290
process() method 294
Java community process 3, 5, 312
Java data objects (JDO) 196
Java development kit (JDK) 37
JavaDoc 53
Java foundation classes (JFC) 231
Java-GNOME project 231
Java media framework (JMF) 587–589
Java messaging services (JMS) 244, 315, 587–589
Java 2 development kit 231
Java 2 enterprise edition (J2EE) platform description of 54–55
implementation based on 62
Java 2 microedition (J2ME) 226, 284
connected device configuration 306
carded limited device configuration 306
elements of 306
event handling 308
mobile information device profile 306, 308
screens 308
Java naming and directory interface (JNDI) 31, 196
data connectors 196
description of 577–581
Java reflection 506
Java remote method protocol (JRMP) 57
JavaScript
code hidden from users 231
description of 230
extensions 234
reusable 230
Java server pages 44
JavaSpaces 464–466
Java specification request (JSR) 160, 312
Java speech application programming interface (JSAPI) 133, 339
markup language 341–342, 343, 344
Recognizer 349
Java speech grammar format (JSGF) 349–350
Java virtual machine (JVM) 53, 72, 82, 226, 286, 468
Java Web start 32, 325
Jini
description of 463–466
JXTA and 466
JXTA
advertisements 451
code 449
cOnfiguration 455–463
data retrieval process 462
definition of 449
features 430
Jini and 466
peer groups 449, 451
pipes 450
protocols 449
JxtaPropagate 454
JxtaUnicast 454
JxtaUnicastSecure 454
K
Knowledge base (KB) of CycL language 168–169, 249
Knowledge connector
application 432
audio factory 411–412
components of 392, 595
Cyc responses 412
formatting and presentation layers 393
goals of 392
object model diagram 393
privilege-based access to knowledgebase 432
programming interface for 392
ServiceConnector class 394
video factory 411–412
Knowledge driven architecture (KDA) 46
Knowledge engine
description of 388
software integration with 390
Knowledge integration 388
Knowledge management 158
KnowledgeService 174
Knowledge technologies
introduction to 151
semantic web 152
software connected with 390
topic maps 136–160
Knowledge workers 389, 391
L
Layered client–server architecture 43
Layout managers 517
Leaders 2
delegation of responsibilities 3
solutions created by 3
Legacy systems 199
Letter-to-sound (LTS) rules 140
Lexicon 139
Lightweight data application protocol (LDAP) 195
Local area networks, wireless 237, 238
Long 476
M
main() method 82, 100, 119, 147
Management style 2
Managers, “pure” 2
Memory cards 287
Messaging systems 201–203
Messenger class 324–326
Index

Messenger object 321
Metadata 382
Metadata repository (MR) 160
Microprocessor cards 286
Microsoft Intermediate Language binary code 65
Microsoft Outlook 247
Microsoft Paint 29
Microsoft Speech Software Development Kit (SDK) description of 351
Emph tag 353
Pitch tag 353
Rate tag 352
Voice tag 353
Volume tag 352
Microsoft Windows CE 284
Microtheones 170–171
MIDlets definition of 310
methods required 322–324
security 311
Mobile information device profile (MIDP) push data from server to devices 311
Mobile information devices 306
Mobile media application programming interface (MMAPI) 309
Model 34
Model-view-controller (MVC) 34, 43, 225, 248
Module implementation advertisement 453
Module specification advertisement 452
Multimedia, wireless 309
Multimedia messaging service (MMS) 260
Multi-tier architecture 28, 39
example of 30–31
MyJXTA 462

N
namesAndLetters 114, 129
namesAndValues 114
Native 476
Natural languages ambiguity of 164
description of 163–164
expressiveness of 164
Natural language semantics markup language (NLISML) 540
Natural language-to-cyc 189
Natural user interface 338
n-dimensional space 24
.NET 252
New 476
NewsLineApplet class 72
NewsLineComponent 79–88
building of 104
changing code of 126
web information channels in 88–100
nextStepRules 246
notEmpty() method 233

O
Object linking and embedding (OLE) 49–50
Object management group (OMG) 50
Object model diagram (OMD) 317
data services definition of 16, 25
for document services 26
knowledge connector 393
topic maps 138
Object-oriented approach abstractions 23, 24
classes 16, 23
classes/responsibilities/collaborators (CRC) approach 15
core module-level requirements 23
description of 9
developmental steps 10
encapsulation 16, 22
inheritance 16–21
limits of 21
n-dimensional space 24
object polymorphism 20–21
rule of “structural” nature of 14
system-level requirements 23
task requirements 23
Object-oriented architecture component connectors 43
corpus 43
definition of 43
Object-oriented concepts 223
Object-oriented programming (OOP) 138
Object request broker 52
Observer 34
Observer design pattern 45
Off-screen graphics 76
Online collaborative services description of 7
limitations 7
members 7
role-based privileges 7
security 7
Ontology definition of 151, 152
RDF 153
RDF Schema 153
search systems that use 152
specific 152
structure of 390
<table>
<thead>
<tr>
<th>Index</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ontology inference layer (OIL)</td>
<td>153</td>
</tr>
<tr>
<td>OpenCyc</td>
<td></td>
</tr>
<tr>
<td>act() method</td>
<td>174</td>
</tr>
<tr>
<td>communication options for description of</td>
<td>174–189</td>
</tr>
<tr>
<td>JXTA</td>
<td>174</td>
</tr>
<tr>
<td>knowledge base</td>
<td>235</td>
</tr>
<tr>
<td>in knowledge container architecture</td>
<td>189</td>
</tr>
<tr>
<td>KnowledgeService</td>
<td>174</td>
</tr>
<tr>
<td>registerObject() method</td>
<td>174</td>
</tr>
<tr>
<td>ServiceConnector class</td>
<td>174</td>
</tr>
<tr>
<td>open() method</td>
<td>196</td>
</tr>
<tr>
<td>OpenOffice</td>
<td>254</td>
</tr>
<tr>
<td>Open source communities</td>
<td>5</td>
</tr>
<tr>
<td>Optical memory cards</td>
<td>287</td>
</tr>
<tr>
<td>Oracle application server (OAS)</td>
<td>195</td>
</tr>
<tr>
<td>Orthogonal frequency-division multiplexing (OFDM)</td>
<td>261</td>
</tr>
<tr>
<td>Package</td>
<td>477</td>
</tr>
<tr>
<td>Packet-switched networks</td>
<td>260</td>
</tr>
<tr>
<td>paint()</td>
<td>77</td>
</tr>
<tr>
<td>PARAM tags</td>
<td>72</td>
</tr>
<tr>
<td>parseXML() method</td>
<td>197, 201</td>
</tr>
<tr>
<td>PauseGenerator</td>
<td>322–324</td>
</tr>
<tr>
<td>Peer advertisement</td>
<td>451</td>
</tr>
<tr>
<td>Peer discovery protocol (PDP)</td>
<td>449</td>
</tr>
<tr>
<td>Peer information protocol (PIP)</td>
<td>449</td>
</tr>
<tr>
<td>Peer resolver protocol (PRP)</td>
<td>449</td>
</tr>
<tr>
<td>“Peers” in JXTA</td>
<td>248, 261</td>
</tr>
<tr>
<td>Personal digital assistant (PDA)</td>
<td>449</td>
</tr>
<tr>
<td>Phraser</td>
<td>138</td>
</tr>
<tr>
<td>Pipe(s)</td>
<td></td>
</tr>
<tr>
<td>definition of JxtaPropagate</td>
<td>450</td>
</tr>
<tr>
<td>JxtaUnicast</td>
<td>454</td>
</tr>
<tr>
<td>JxtaUnicastSecure</td>
<td>454</td>
</tr>
<tr>
<td>Pipe advertisement</td>
<td>454</td>
</tr>
<tr>
<td>Pipe binding protocol (PBP)</td>
<td>450</td>
</tr>
<tr>
<td>PitchMarkGenerator</td>
<td>139</td>
</tr>
<tr>
<td>playAct() method</td>
<td>235</td>
</tr>
<tr>
<td>playScenario() method</td>
<td>415, 423</td>
</tr>
<tr>
<td>Polymorphism</td>
<td>20–21, 465</td>
</tr>
<tr>
<td>Port 80</td>
<td>223</td>
</tr>
<tr>
<td>PostLexicalAnalyzer</td>
<td>138</td>
</tr>
<tr>
<td>Predicates</td>
<td>169</td>
</tr>
<tr>
<td>Private</td>
<td>477</td>
</tr>
<tr>
<td>Private branch exchange systems</td>
<td>260</td>
</tr>
<tr>
<td>Private data</td>
<td>127</td>
</tr>
<tr>
<td>Procedure-based architecture</td>
<td>44</td>
</tr>
<tr>
<td>process() method</td>
<td>290, 294</td>
</tr>
<tr>
<td>Product design</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>P</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Project management</td>
<td></td>
</tr>
<tr>
<td>description of</td>
<td>2</td>
</tr>
<tr>
<td>product design and</td>
<td>3</td>
</tr>
<tr>
<td>Protected</td>
<td>477</td>
</tr>
<tr>
<td>Protected data</td>
<td>127</td>
</tr>
<tr>
<td>Proxy</td>
<td>34</td>
</tr>
<tr>
<td>Public</td>
<td>477</td>
</tr>
<tr>
<td>Publish/Subscribe mechanism</td>
<td>45</td>
</tr>
<tr>
<td>“Pure” management</td>
<td>3</td>
</tr>
<tr>
<td>Push access protocol (PAP)</td>
<td>284</td>
</tr>
<tr>
<td>Push proxy gateway (PPG)</td>
<td>284</td>
</tr>
<tr>
<td>Quality assurance engineer</td>
<td>10</td>
</tr>
<tr>
<td>Quality assurance managers</td>
<td>195</td>
</tr>
<tr>
<td>Query element</td>
<td>420</td>
</tr>
<tr>
<td>Radiofrequency engineers</td>
<td>259</td>
</tr>
<tr>
<td>Rapide</td>
<td>47</td>
</tr>
<tr>
<td>Rational Unified Process</td>
<td>5</td>
</tr>
<tr>
<td>RDF</td>
<td>153</td>
</tr>
<tr>
<td>RDF Schema</td>
<td>153</td>
</tr>
<tr>
<td>readXMLObject() method</td>
<td>300</td>
</tr>
<tr>
<td>readTextFile()</td>
<td>119</td>
</tr>
<tr>
<td>RecordHandler class</td>
<td>319, 328</td>
</tr>
<tr>
<td>Record management system (RMS)</td>
<td>310</td>
</tr>
<tr>
<td>Record stores</td>
<td>310–311</td>
</tr>
<tr>
<td>ReferenceService class</td>
<td>207, 209, 213</td>
</tr>
<tr>
<td>registerObject() method</td>
<td>174</td>
</tr>
<tr>
<td>Relation</td>
<td>136</td>
</tr>
<tr>
<td>Relational database management systems (RDBMS)</td>
<td>196</td>
</tr>
<tr>
<td>Remote authentication dial in user service (RADIUS)</td>
<td>262</td>
</tr>
<tr>
<td>Remote method invocation (RMI)</td>
<td>57</td>
</tr>
<tr>
<td>Remote procedural call (RPC)</td>
<td>223</td>
</tr>
<tr>
<td>Remote scenario processing</td>
<td>281</td>
</tr>
<tr>
<td>Rendezvous protocol (RVP)</td>
<td>450</td>
</tr>
<tr>
<td>Representational state transfer (REST)</td>
<td>43</td>
</tr>
<tr>
<td>Return</td>
<td>477</td>
</tr>
<tr>
<td>Reusability</td>
<td>33</td>
</tr>
<tr>
<td>Rich clients</td>
<td>130</td>
</tr>
<tr>
<td>description of</td>
<td>255</td>
</tr>
<tr>
<td>OpenOffice features added to as peer</td>
<td>254</td>
</tr>
<tr>
<td>Robots</td>
<td>199</td>
</tr>
<tr>
<td>Roles, in Cycl language</td>
<td>171–172</td>
</tr>
<tr>
<td>Rule grammar</td>
<td>349</td>
</tr>
<tr>
<td>run() method</td>
<td>73, 325</td>
</tr>
<tr>
<td>Runnable</td>
<td>120</td>
</tr>
<tr>
<td>Sbyte</td>
<td>471</td>
</tr>
<tr>
<td>SBYTE</td>
<td></td>
</tr>
<tr>
<td>ScenarioPlayer class</td>
<td>233–237</td>
</tr>
</tbody>
</table>
Index

Scenarios
description of 13
XML. See Scenario
Security monitoring 586
Segmenter 138
select() method 290, 294
Semantic web 152
send() method 324
Service-based architecture 138
ServiceComponent
Service component objects 113
ServiceConnector
ServiceConnector class 211, 317, 394
act() method of 318
ServiceController interface
components 101
creation of 107
dispatch() method 99
OpenCyc 174
ServiceControl object
service() method 91
Service-oriented architecture (SOA)
Servlet 253
ServletPerformer 394
setGroupRoles() method 299
Shared information systems 46
Shell script 84
Short
Short message service (SMS)
259, 262–263, 336
from limitation of 313
port numbers restricted to 314
wireless messaging with 312
Short messaging service center (SMSC) 263
Simple API for XML programming 103
Simple object access protocol (SOAP) 64, 223
Simple secure network (SSN) 262
Simulators 268
Single-user architecture 28
Six Sigma 4, 3
Skeletone 52
Slots
161–163
Smart cards
cost of 287
definition of 286
memory cards 287
microprocessor cards 286
optical memory cards 287
Sofology 387
Software architect 10
Software architecture 42
SpecificServiceComponent
Speech application language tags (SALT) 360–361
Speech recognition grammars (SRGS) 366
Speech Recognition System
in C++ 589
components of 363–366
description of 132
Index

608

limitations 227
local services for 247
“pure” 248
reusable 234
smarts added to 234
Thinlet 226
ultra light client 227
value object assembler pattern added to 228
Thinlet 226, 320
mobile information device profile 226
Third-generation (3G) 260
Third-generation (3G) partnership project 260
Throw 480
Throws 480
Tiered architecture 43
Time division multiple access (TDMA) 239
Time-to-live (TTL) 466
Tokenization 137
Tokenizer 137
TokenToWords 137
Tool Provider 10
Topic
definition of 136
occurrences of 136
scope of 137
Topic association 136
Topic maps
definition of 136–160
object model diagram 138
object-oriented modeling 137
XML for 157, 158
Transactional capabilities application part (TCAP) 263
Transient 481
translate element 418
Truth maintenance system (TMS) 168
Try 482
Unix 29
update() 77
URL 89, 244
creation of
Use cases of 12, 315–317
User audio–video interface (UserAVI) 320, 321
UserAVI class 395, 414
User interface (UI)
description of 36
graphical. See Graphical user interface
User’s view of architecture 47
UtteranceProcessors 135, 136, 137
V
Value object assembler (VOA) design pattern 225, 228
View 34
VirtualCurrency applet
application-specific methods 295
data structures 291
description of 289
review of 305
standard methods 292–294
writing of 291
Virtual team 8
VoiceComponent
ControlComponent and
main() method 140–143
Voice components
coding of 140
description of 133
VoiceXML
architecture 366
description of 364
development of 364
documents 366–368, 369–378
document server 366
ECMAStories in documents 380
elements
“forward mail” service request 373
“new mail” service request 373
speech recognition systems 365–366
transfer operation code 372
VoiceXML interpreter
description of 380
metadata 382
resources 382
self-evaluation by 382
Void 482
Volatile 482
W
Web browsers 98
Droplet versus 227, 234
infrastructure 226
limitations of 226
Web information channels 88–100

Cambridge University Press
Jeff Zhuk
Index
More information

© Cambridge University Press www.cambridge.org
Index

WebNewsContainer 98, 109
Web server 29
Web services 39
register of 64–65
universal description, discovery, and integration (UDDI) 540
vendors that support 223
wireless application protocol devices and XML 544
Web services description language (WSDL) 45, 64, 223
While 483
while (true) loop 75
Wide area networks, wireless 257
Windows Forms 250, 531
Wireless application protocol (WAP) basis for 264
business advantages 263
business infrastructure connection 268
communications protocol 263
definition of 263–265
development team 268
devices 267, 284
devices, memory limitation for 278
e-mail with attachments 279
gateway 265, 335
oversized data 279
programming pages 264, 270
Wireless application protocol (WAP) push 281
Wireless application server (WAS) 335
Wireless clients 258
Wireless fidelity 261
Wireless internet service providers (WISP) 258
Wireless local area networks (WLAN) 257, 258, 261
access points 261
securing of client’s traffic 261
Wireless markets multimedia 309
types of 257
Wireless markup language (WML) 103, 226
“buildPages” module 279
description of 264
presentation factory 269
programming 270
template 277
transactions secured with Wireless messaging client application using 314
with short message service 312
Wireless messaging application programming interface 312
Wireless personal area networks (WPAN) 257
Wireless telephony application (WTA) 263
Wireless wide area networks (WWLAN) 257
Wrapper 225
XML 544
Elements 544
Web services 541
XML descriptor description of document services and example 197
presentation definition 209
Thinlet and validation rules 226
XML for topic maps 157, 158
XML scenario act() method 415
acts of 393
C# 513
elements of 416
type of 421
executable acts of 425–426
execution order of 423
playing of 395, 413
playScenario 423
service action executed in user authentication 415
writer of 416
XML style sheet language for transformations 342

© Cambridge University Press www.cambridge.org