

# Index

## A

- Absolute location path, 88
- Access, 9
- Accessibility, 6
- ACORD vocabulary, 41
- Adapter, 193
- Agility, 6
- Ancestor scopes, 133
- Anonymous data types, 56
- Apache Software Foundation, 4, 195, 228
- Application, 27
- Application programming interface (API), 227
- Assign activity, 158–165, 268–269
  - attributes of each copy element, 165
  - expression as target, 163–164
  - literals, 160–161
  - partner links, 164–165
  - types and namespaces, 161–163
- Assumptions, identification of, 32
- Asterisk (\*) node test, 100
- Asynchronous communication, 17
- Attribute node, 86, 98
- Attributes, 44–45, 165, 215, 264
- Authentication, 19
- Authorization, 12, 19
- Axis specification, 96–100
  - ancestor, 97
  - ancestor-or-self, 97
  - attribute nodes, 98

- categorization of, 100
- child, 96, 97
- descendant, 98
- descendant-or-self, 98
- examples with descendants and siblings, 106–109
- following, 98
- following-sibling, 98
- forward axis, 97
- namespace, 99–100
- parent, 98
- preceding, 98
- preceding-sibling, 98
- reverse axis, 97, 107–108
- self, 98

## B

- Base data types, 56
- Basic activities, 119, 131, 157, 268–277
- Binary-exchange services, 5–6
- Binding, 11, 200–201
- Binding element, 75–78
- Binding style, 76
- Binding template, 83
- Body element, 79, 81
- Boolean expressions, 89, 103, 111–112
- Boolean properties, 227

- BPEL. *See* Business Process Execution Language (BPEL)
- BPEL Extensions for Sub-Processes, 124
- BPEL4People, 124
- BPELJ, 124
- BPEL process, 85
- Branches, 122
- Business entity, UDDI, 83
- Business process, analyzing
  - approval of design, 29
  - facilitated sessions, 29
  - one-on-one information gathering, 29
  - refining of plans, 29
  - service points, 29
  - software adapters, 29
  - step-by-step procedure for, 28–30
- Business Process Execution Language (BEPL), xvii, 72, 85, 119–155
  - abstract process, 123
    - activities, 119, 157–192
    - assign, 158–165, 268–269
    - basic, 119, 131, 157, 268–277
    - compensate, 270–271
    - compensateScope, 271–272
    - doXSLTransform, 186–192, 285–286
    - empty, 272
    - event handlers, 181–183, 288–289
    - exit, 272
    - extensionActivity, 273
    - flow, 278–279
    - forEach, 175–177, 279–280
    - fromParts, 166–170
    - getVariableProperty, 286
    - if, 280
    - invoke, 170–172, 273–274
    - iterative construction, 187, 190–192
    - loops, 177–178
    - message exchanges, 134–137, 184–186, 291
    - onAlarm event, 180
    - onMessage event, 179–180
    - options for, 122
    - pick, 178–180, 281–282
    - receive, 173, 274–275
    - repeatUntil, 178, 282
    - reply, 173–174, 275
    - rethrow, 276
    - scope, 283–284
    - sequence, 284
    - single transformation, 187–189
    - start, 157–158
    - structured, 119, 132, 157, 278–285
    - throw, 276
    - toParts, 166–170
    - validate, 277
    - wait, 180–181, 277
    - while, 178, 285
  - BPELVariableName, 264
  - BPELVariableNames, 264
    - compensation handling, 122, 133, 144–146, 287
    - concurrency, 122, 146–155
      - advanced, 151–155
    - correlation, 122
    - correlation sets, 138–140, 287
    - deadlines, 267
    - defined, 119
    - durations, 265–266
    - engine, 41, 122
    - executable process, 122
    - extending, proposals for, 124
    - extensions to WSDL, 296–298
    - fault handling, 122, 141–144, 290
    - file structure, 130–131
    - functions, 285–286
    - imports, 290–291
    - join conditions, 153–155
    - NCName, 51, 265
    - partner links, 121, 124, 125–126, 137–138, 164–165, 291–292
    - partner link types, 124, 296–297
    - partner service, 119
    - process, 85, 119–122, 190–192, 292–293
      - excerpt from, 120–122
    - property, 125, 126–130, 297
    - property aliases, 125, 126–130, 298
  - QName, 265
  - QName-list, 265
  - running time, 119
  - sample flow activity outline, 148
  - scopes, 132–134

- standard-attributes, 264
- standard-elements, 264
- termination handling, 133, 146, 294
- transition conditions, 152–153
- use of WSDL, 124–130
- variables, 121, 294–296
- Version 2.0*, reference guide to, 263–298

Business services, 4, 83

## C

- Callback, 17
- Callback operation, 17–18
- Cast, 53
- Category bag, 84
- Change-tracking, 230–231
- Child element, 44, 96, 97, 132
- Choreography, 21
- Clarity, xv
- Coarse-grained services, 6
- Comment node, 86
- Comment() node test, 101
- Commit changes, 21
- Compensate activity, 270–271
- CompensateScope activity, 271–272
- Compensating services, 21–22
- Compensation handling, 122, 133, 144–146, 287
- Complex data types, 52–53, 56–57
- Component(s), 202–214
  - and composite compared, 199
  - defined, 195
  - property, 195, 204, 213–214
  - reference, 195, 205, 213–214
  - service, 203–204
  - type, 211–212
- ComponentType file, 212–213
- Composite(s), 214–226
  - attributes in, 215
  - and component compared, 199
  - constraining type, 226
  - defined, 196, 214
  - higher- and lower-level, 223–224
  - inclusion, 225
  - property, 217–218, 224
  - reference, 218–220, 224
  - service, 215–217, 224

- wires, 220–223

Concurrency, 122, 146–155
 

- advanced, 151–155
- join condition, 153–155
- optimistic, 237
- transition condition, 152–153

Confidentiality, 20, 201, 203

Configuration setting, 22

Constraining type, 226

Content, 12, 44

Content model, 54

Context, 91–93

Context diagram, 33

Context node, 92

Context position, 93

Contract
 

- defined, 11
- Quality of Service (QoS), 11
- service interface, 11

Conversation, 23, 24, 202

Correlation, 24–25, 122

Correlation sets, 138–140, 287
 

- errors, effect of, 140
- initiation attributes, 138, 139–140
- join value, 140
- no value, 139–140
- request-response value, 140
- request value, 140
- response value, 140
- yes value, 140

## D

- Data Access Service (DAS), 5, 236–238
- Data graph, 228–233
  - change-tracking, 230–231
  - defined, 228
  - inverse integrity, 231–233
  - relationship to the data in a data source, 230
  - terminology, 233
- Data Objects, 227
- Data type(s), 51–53
  - anonymous, 56
  - base, 56
  - complex, 52–53, 56–57
  - derived, 52

Data type(s), *continued*

- global, 55
  - group declaration, 57–58
  - local, 56
  - primitive, 52
  - reuse, 62
  - set of facets, 56
  - simple, 52, 56
  - strings, 54
  - variable, 53
- Dead path elimination, 153
- Deep copy, 238
- Default namespace, 48, 49, 61
- Definitions start-tag element, 70–71
- Denial-of-service attacks, protection from, 20
- Derived data type, 52
- Descendant elements, 44, 98, 132
- DOCTYPE declaration, 88
- Document, 228
- Document category, service interface, 13
- Document-literal wrapped message format, 71
- Document Type Definition (DTD), 41, 88
- doXSLTransform activity, 186–192, 285–286
- Duplicate logic, 2
- Dynamic definition, 234–235, 244–246

**E**

## Element(s)

- attributes, 44–45, 165
- binding, 75–78
- body, 79, 81
- child, 44, 96, 97, 132
- complex content schema, 60
- content, 12, 44
- descendants, 44, 98, 132
- empty, 45
- envelope, 78
- header, 79
- instance, 55
- message, 73–74
- parent, 44, 98, 133
- portType, 74–75
- sequencing, 58–59
- simple content Schema, 59–60
- SOAP, 78–79

- standard-, 264
- Elementary access details, 11
- Element node, 86
- Empty activity, 272
- Empty element, 45
- Empty set, 88, 90
- Encoded use value, 76
- Encryption key, 251
- Encryption software, 251
- Endpoint, 22–23, 78
- Endpoint reference, 125
- Enterprise Generation Language (EGL), 5, 11
- Enterprise JavaBean stateless session bean (EJB SLSB), 193
- Enterprise Service Bus (ESB), 201
- Envelope element, 78
- Event handlers, 181–183, 288–289
- Exit activity, 272
- Explicit header, 77
- Expression language, 123
- Extensible Markup Language. *See* XML (Extensible Markup Language)
- Extensible Markup Language Stylesheet Language Transformation (XSLT). *See* XML Stylesheet Language Transformation (XSLT)
- ExtensionActivity, 273
- Extension assign operation, 159

**F**

- Fault handling, 122, 141–144, 290
- exiting in response to a fault, 144
  - inner life, 143–144
  - selection at run time, 142–143
- Fire-and-forget message exchange pattern, 16
- Flexibility, 1, 77
- Flow activity, 278–279
- forEach activity, 175–177, 279–280
- Foreign key, 230
- Formatting, 11, 12, 76
- fromParts activity, 166–170
- Function calls, 86
- Function invocation, 2

**G**

GetVariableProperty function, 286  
 Global data types, 55  
 Group declaration, 57–58

**H**

Header block, 79, 80–81  
 Header element, SOAP, 79  
 Help-desk application, 193  
 Helper interfaces, 238–239  
 Highlight Insurance (fictitious company), 27–38  
   assumptions, communicating in writing, 32  
   business process, analyzing, 28–30  
   current process, identifying problems with, 31  
   expansion, plan for, 28  
   independent agencies and, 28  
   insurance application, 27  
   isolating services, 33–34  
   overview of, 28  
   policy application, creating, 36–38  
   policy request, 27  
   quote, 27  
   quote application, creating, 34–36  
   quote request, 27  
   sales process at, 30  
 Hypertext Markup Language (HTML), 9  
 Hypertext Transfer Protocol (HTTP), 5, 11, 200

**I**

Identifier bag, 84  
 if activity, 280  
 IFX vocabulary, 41  
 Imagination, xv  
 Implementation code  
   defining services in, 206–208  
   mapping of properties to, 208–209  
   mapping of references to, 209–211  
 Implementation, defined, 198–199  
 Implementation policy, 201, 203  
 Implicit header, 77  
 Import declaration, 62  
 Imports, 290–291  
 Inbound message activities (IMAs), 173, 275  
 Include declaration, 62

Information hiding, xv  
 In-only message exchange pattern, 16  
 In-out message exchange pattern, 16  
 Instance attributes, 55  
 Instance document, 55, 62–63  
 Instance elements, 55  
 Integer properties, 227  
 Integers, 53  
 Integration services, 4  
 Integrity, 20, 231–233  
 Intent, 201  
 Interaction policy, 201, 203  
 Intermediaries, 10  
 Internet Engineering Task Force (IETF), 47  
 Interoperability, 3  
 Inverse integrity, 231–233  
 Invoke activity, 170–172, 273–274  
 Iterative construction, 187, 190–192

**J**

Java API for XML Processing (JAXP), 86  
 Java EE Connector Architecture (JCA), 200  
 Java Message Service (JMS), 11, 194, 200  
 Java Remote Method Invocation over Internet  
   Inter-ORB Protocol (JMI/IIOP), 200  
 Join condition, 153–155

**L**

Latency period, 19  
 Literals, 86, 160–161  
 Literal use value, 76  
 Load balancing, 22  
 Local data types, 56  
 Local name, 51  
 Location, 11  
 Location path, 86, 88  
 Location step, 88  
   abbreviations, 104–105  
   axis specification, 96–100  
   node tests, 96, 101–103  
   parts of, 96–104  
   predicates, 96, 103–104  
   summary of, 108–109  
 Logging, 201, 203

- Logic, 31
  - duplicate, 2
- Log information, 9
- Loops, 177–178
- Loose coupling, 1, 6, 13–14
  
- M**
- Mapping, 205
  - of maps, 125
  - of properties to implementation code, 208–209
  - of references to implementation code, 209–211
- Message(s)
  - content, 12
  - document-literal wrapped format, 71
  - element, 73–74
  - exchange patterns, 12, 16–18, 67
  - exchanges, 134–137, 184–186, 291
  - formatting, 11, 12, 76
  - queues, 14, 193
  - reformatting, 10, 22
  - rerouting, 10
  - runtime update of content/destination, 22
  - type, 134, 294
  - unformatting, 11
- Message exchange patterns (MEPs), 12, 16–18
  - callbacks, 17–18
  - notification, 17
  - one-way, 18
  - request-response, 16
  - solicit-response, 17
  - supported by Web Services Description Language (WSDL), 67
  - synchronous and asynchronous communication, 17
- Metadata, 235
- Migration
  - of existing applications, 2, 7
  - incremental, 7
- Mixed content, 44
- Model, 84
- Modular code, xvi
- Multi-valued reference, 205

**N**

- Namespace(s), 46–51, 161–163
  - axis specification, 99–100
  - default, 48, 49, 61
  - defined, 46
  - example and terminology, 50–51
  - identifiers, 47–48
  - local name, 51
  - NCName, 51, 265
  - node, 86
  - non-default, 48, 50
  - purpose of, 46–47
  - QName, 51, 265
  - qualification and declarations, 48–49
  - rules, 49–50
  - in scope, 49
  - second declaration, 61
  - target, 61
    - in XML Schema definitions (XSDs), 60–62
- NCName, 51, 265
- Negotiation, 12
- Nested scopes, 132
- Node() node test, 102
- Nodes, 86–89, 98
- Node tests
  - asterisk (\*), 100
  - Boolean logic in, 111–112
  - comment(), 101
  - defined, 100
  - node(), 102
  - processing instruction(), 102–103
  - text(), 101–102
- Non-default namespace, 48, 50
- Non-repudiation, 20
- Notification broker, 256
- Notification message exchange pattern, 17
  
- O**
- OASIS. *See* Organization for the Advancement of Structured Information Standards (OASIS)
- Object definition, 234–235
  - dynamic, 234–235, 244–246
  - static, 234, 241–244

- onAlarm event, 180
  - onAlarm handler, 182–183
  - onEvent handler, 181–182
  - One-way message exchange pattern, 16, 67
  - onMessage event, 189–180
  - Open Grid Forum, 3
  - Open Service Oriented Architecture (OSOA)
    - collaboration, 194, 228
    - SCA Assembly Model Specification*, 202, 218, 226
    - SCA Client and Implementation Mode Specification for WS-BPEL*, 209
    - SCA Policy Framework*, 202
    - Service Data Objects for Java Specification Version 2.1.0*, 228, 238
  - Open source, defined, 3
  - Open standards, 2–4
  - Operands, 86
  - Optimistic concurrency, 237
  - Orchestration, 20
  - Organization for the Advancement of Structured Information Standards (OASIS), 3, 66, 124
  - OSOA. *See* Open Service Oriented Architecture (OSOA) collaboration
  - Out-in message exchange pattern, 17
  - Out-only message exchange pattern, 17
  - Overwriting, 237
- P**
- Parallel streams, 122
  - Parent element, 44, 98, 133
  - Partner links, 121, 124, 125–126, 137–138, 164–165, 291–292
  - Partner link types, 124, 296–297
  - Partner service, 119
  - Peer scopes, 132, 133
  - Pick activity, 178–180, 281–282
  - Policies, 201
  - Policy set, 201
  - Portal, 9
  - Portlet, 9
  - portType element, 74–75
  - Positional predicate, 93
  - Predicate, 89
  - Presentation services, 8–9
  - Primitive data type, 52
  - Process, 292–293
  - Process definition, 130–131
  - Processing-instruction node, 86
  - Processing instruction() node test, 102–103
  - Processing instructions (PIs), 45–46
  - Process instance, 24
  - Profile requirements, 247
  - Profiles, 247
  - Property(ies), 125, 126–130, 297
    - component, 195, 213–214
    - composite, 195, 204, 213–214
    - mapping of to implementation code, 208–209
  - Property alias, 125, 126–130, 298
  - Publish-and-subscribe pattern, 255, 256
  - Publisher assertion, 83
- Q**
- QName, 51, 265
  - QName-list, 265
  - Quality of Service (QoS), 12, 18–22, 65, 247
    - capabilities in SCA, 194
    - contract, 11
    - reliability guarantees, 18, 19
    - runtime update of message
      - content/destination, 19, 22
    - security mechanisms, 19–20
    - service coordination, 19, 20–22
    - specifying within Web Services Description Language (WSDL), 77
    - transaction control, 19, 21
  - Query language, 123
  - Quotation marks (for delimiting strings), 90
- R**
- Receive activity, 173, 274–275
  - Reference(s)
    - component, 195, 205, 213–214
    - composite, 218–220, 224
    - endpoint, 125
    - mapping of to implementation code, 209–211
    - multi-valued, 205
  - Reformatting, 10, 22
  - Relative location path, 88

RELAX NG, 41  
Reliability guarantees, 18, 19  
Remote procedure call (RPC), 13, 228  
repeatUntil activity, 178, 282  
Reply activity, 173–174, 275  
Repository, 83  
Request-response message exchange pattern, 16, 67  
Request-response value, 140  
Request value, 140  
Rerouting, 10  
Response value, 140  
Restriction, 247  
Result document, 186  
Rethrow activity, 276  
Reusability, 6  
Reverse axis, 97, 107–108  
Rollback changes, 21  
Root element, 87  
Root node, 87  
Runtime, xviii, 9–10, 119  
Runtime response, 5

## S

SCA. *See* Service Component Architecture (SCA)  
SCA assembler, 196  
SCA Domain, 200  
Schematron, 41  
Scope(s), 132–134  
    activity, 283–284  
    ancestor, 133  
    defined, 132  
    descendants, 132  
    nested, 132  
    peer, 132, 133  
SDO. *See* Service Data Objects (SDO)  
Second namespace declaration, 61  
Security mechanisms, 9, 10, 19–20, 251  
Self-documenting language, 41  
Sequence activity, 284  
Sequencing element, 58–59  
Service(s), x, 8–9, 11–25  
    availability, 14  
    binary-exchange, 5–6  
    business, 4, 83  
    coarse-grained, 6  
    compensating, 21–22  
    component, 203–204  
    composite, 215–217, 224  
    contract, 11  
    data-access, 5  
    defined, 197–198  
    defining in implementation code, 206–208  
    elementary access details, 11  
    endpoints, state, and correlation, 22–25  
    functions of, 1–2  
    integration, 4  
    location, 2  
    loose coupling, 13–14  
    message exchange patterns (MEPs), 16–18  
    partner, 119  
    Quality of Service (QoS), 18–22  
    service implementation, 11  
    service registry, 15  
    stateful, 23  
    stateless, 23  
    Web, 5  
Service Component Architecture (SCA), xvii, 85, 193–226  
    assembly project phase, 196  
    benefits of, 194  
    bindings, 200–201  
    components, 202–214  
    componentTypefile, 212–213  
    component types, 211–212  
    composites, 214–226  
    defined, 193  
    defining services in implementation code, 206–208  
    deployment project phase, 197  
    development project phase, 195  
    Enterprise Service Bus, 201  
    implementation, defined, 198–199  
    implementation languages supported by, 194  
    mapping of properties to implementation code, 208–209  
    mapping of references to implementation code, 209–211  
    open-source implementation of, 3–4



- policies and support for conversations, 201–202
- properties and references, 213–214
- Quality of Service (QoS) capabilities, 194
- service, defined, 197–198
- standards proposals, 194–195
- tasks made possible by, 193–194
- Service coordination, 20–22
- Service Data Objects (SDO), xviii, 85, 227–246
  - advanced capabilities in Java, 246
  - annotations, 235–236
  - change-tracking, 230–231
  - code details, 238–246
  - Data Access Service (DAS), 236–238
  - data graph, 228–233
  - defined, 227
  - example code, 239–246
  - helper interfaces, 238–239
  - introduction to, 227–228
  - inverse integrity, 231–233
  - object definition, 234–235
  - open-source implementation of, 3–4
  - standards proposals, 228
- Service element, 78
- Service implementation, 11
- Service instance, 23, 24
- Service interface, 11, 12
  - defined, 18
  - document category, 13
  - remote procedure call (RPC) category, 13
  - Web Service Description Language (WSDL), 70–75
- Service Level Agreement (SLA), 15–16, 19, 83
- Service points, 29
- Service registry, 15
- Service requests, 9
- Service-oriented application
  - defined, 4–5
  - defining, steps in, 33–34
  - structure of, 4–5
- Service-oriented architecture (SOA)
  - benefits of, 6
  - business implications, 6–8
  - correlation, 24–25
  - criteria for, 7
  - defined, xvii, 1
  - endpoint, 22–23
  - migration of existing applications, 2, 7
  - open standard technologies, 2–4, 65–84
    - Web Services Description Language (WSDL), 65, 66–78
    - SOAP, 65, 78–82
    - Universal Description, Discovery and Integration (UDDI), 65, 82–84
  - presentation services, 8–9
  - proposed standards, 247–258
  - runtime products, 9–10
  - service-oriented application, structure of, 4–5
  - state, 23–24
  - transition to, xvi
  - user concerns about, xvi
  - Web and binary-exchange services, 5–6
  - vs. Web services, 7–8
- Shallow copy, 238
- Simple data types, 52, 56
- Simple Object Access Protocol. *See* SOAP (Simple Object Access Protocol)
- Single transformation, 187–189
- SOAP (Simple Object Access Protocol), 5, 65, 78–82
  - body element, 79, 81
  - defined, xvii
  - engine, 80
  - envelope element, 78
  - example, 78–80
  - format, 78–79
  - header block, 79, 80–81
  - header element, 79
  - at runtime, 81–82
- Solicit-response message exchange pattern, 17
- Source, 159
- Standard-attributes, 264
- Standard-elements, 264
- Standard faults, 141
- Standards
  - open, 2–4
  - proposals for, 194–195, 247–258
  - purposes of, 247
- Start activity, 157–158
- State, 23

Stateful service, 23  
State information, 14  
Stateless service, 23  
Static definition, 234, 241–244  
Strings, 53, 90  
Strongly typed computer language, 53  
Structured activities, 119, 132, 157, 278–285  
Synchronization, 147. *See also* Concurrency  
Synchronous communication, 17

## T

Target, 159  
Target namespace, 61  
Termination handling, 133, 146, 294  
Text() node test, 101–102  
Text node, 86  
Thread of execution, 23  
Throughput, 19  
Throw activity, 276  
toParts activity, 166–170  
Traffic flow, 22  
Transaction control, xviii, 19, 21  
Transition condition, 152–153  
Transport protocol, 3, 11  
Types element, 71–73

## U

Unformatting, 11  
Uniform Resource Locator (URL), 47–48  
Universal Description, Discovery and  
Integration (UDDI), 65, 82–84  
registry categories, 83–84  
rules of, 15  
Universal Resource Identifier (URI), 47, 51  
Use value, 76

## V

Validate activity, 277  
Variables, 53, 121, 134–137, 294–296

## W

Wait activity, 180–181, 277  
Weakly typed computer language, 53

Web services, xvii, 5  
*Web Services Business Process Execution  
Language Version 2.0*, 263  
Web Services Description Language (WSDL),  
5, 65, 66–78  
binding element, 75–78  
complexity, 66–67, 75  
defined, xvii, 66  
endpoint reference, 78  
MEPs supported by, 67  
message element, 73–74  
portType element, 74–75  
preferred message format, 71  
Quality of Service (QoS), specifying within,  
77  
service element, 78  
service interface, 70–75  
start-tag, 70–71  
types element, 71–73  
typical definition, outlines of, 67–69  
use of in BPEL, 124–130  
uses of, 66  
XSDs in, 54  
Web Services Distributed Management  
(WSDM), 254, 255  
Web Services for Remote Portlets, 9, 257  
Web Services Interoperability Organization  
(WS-I) Basic Profile, 3, 247  
Web Services Resource Transfer (WS-RT),  
256–257  
WebSphere MQ, 193  
While activity, 178, 285  
World Wide Web Consortium (W3C), 3, 66  
WS-\* (WS Splat), 247–258  
defined, xviii  
Web Services Distributed Management  
(WSDM), 254, 255  
Web Services for Remote Portlets, 257  
Web Services Resource Transfer (WS-RT),  
2567–257  
WS-Addressing, 249  
WS-Agreement, 12, 258  
WS-Atomic-Transaction, 252–253  
WS-BusinessActivity, 253  
WS-Coordination, 254

- WS-Enumeration, 257
- WS-Eventing, 256
- WS-EventNotification, 255–256
- WS-Management, 254, 255
- WS-MetadataExchange, 252
- WS-Policy, 249–250
- WS-ReliableMessaging, 250
- WS-ResourceFramework, 257
- WS-Security, 251–252
- WS-Trust, 251–252
- WS-Unified Management (WS-UM),  
254–255

## X

XML (Extensible Markup Language) 39–63

- attributes, 44–45
- benefits of, 41
- child, 44
- comments, 43
- content, 44
- for data transfer, 5
- declaration, 43
- defined, xvii, 39
- descendants, 44
- document structure, 43–46
- encoding, 43
- engine, 40–41
- mixed content, 44
- namespace, 46–51
- overview of, 40–41
- parent, 44
- processing instructions (PIs), 45–46
- processor, 40–41
- Schema definitions, 51–63
  - in SOA, 41–43
- source, 86
- source order, 88
- stored values, 86

- vocabulary, 40, 41
- XML Path Language. *See* XPath
- XML Schema Definitions (XSDs), 41, 51–62
  - content model, 54
  - data type, 51–53, 55–57
  - data-type reuse, 62
  - import declaration, 62
  - include declaration, 62
  - instance document, 55, 62–63
  - namespaces in, 60–62
  - purpose of, 53–54
  - sequencing, 58–59
  - simple and complex content elements, 59–60
  - structure of, 54–60
  - unqualified setting, 61, 63
- XML Stylesheet Language Transformation (XSLT), 85, 186
- XPath, 85–117
  - absolute location path, 88
  - avoiding errors, 89–91
  - context, 91–93
  - defined, xvii, 86
  - examples, 94–96
  - expressions, 109–112
  - functions, 113–117
    - returns a Boolean, 114
    - returns a number, 116–117
    - returns a string, 114–115
  - location path, 86, 88
  - location step, 88, 96–104, 108–109
  - nodes, 86–89
  - node set, 88, 110–111
  - numeric and Boolean operators, 113
  - predicates, 89, 103–104
  - relative location path, 88
  - setup for practice in, 259–262
  - uses of, 86
  - variables, 135–137
- XQuery 2.0, 85