

Index

A

about this book

- audience, 6
- conventions, 19
- objectives, 4
- organization, 7-19
- what's not covered, 6-7

abstract description. *See* WSDL, abstract description

abstraction

- of application logic. *See* service abstraction layers
- of business logic. *See* service abstraction layers
- layers. *See* service abstraction layers
- with J2EE, 686
- with .NET, 702
- with services. *See* service-orientation principles, service abstraction
- with SOA. *See* service abstraction layers

ACID transactions, 187-188

active intermediary.

- See* intermediaries, active
- See* SOAP, active intermediary

Active Server Pages. *See* .NET platform, ASP.NET

activities. *See* service activities

addressing

- explained, 220-228
- See also* WS-Addressing

advertisement. *See* service descriptions, advertisement

agile strategy. *See* SOA delivery strategies, agile

APIs

- See* .NET platform, APIs
- See* J2EE platform, APIs

application architecture, 86-87

application logic (SOA), 280-283

application programming interfaces. *See* APIs

application service

- case study examples. *See* case study examples, service-oriented design examples (service design)
- deriving candidates, 399-415, 440
- design process. *See* service design, application service design step-by-step process description
- explained, 337-339, 718-719
- in service layer configurations, 347-353

application service layer
 case study example. *See* case study examples, application service layer
 explained, 335-341, 522

architecture
 application. *See* application architecture
 client-server. *See* client-server architecture
 component-based. *See* distributed architecture
 defined, 86-88
 distributed Internet. *See* distributed architecture
 hybrid Web service. *See* Web services, non-SOA architecture
 enterprise. *See* enterprise architecture
 service-oriented. *See* SOA

ASMX. *See* .NET platform, ASP.NET

ASP.NET. *See* .NET platform, ASP.NET

ASP.NET Web Forms, 689

ASP.NET Web Services. *See* .NET platform, ASP.NET

ASP.NET worker process. *See* .NET platform, ASP.NET worker process

atomic activity. *See* service modeling, primitive business activity

atomic transactions
 explained, 186-193
See also WS-AtomicTransaction

attachments. *See* SOAP, attachments

authentication. *See* security, authentication

authorization. *See* security, authorization

autonomy
 in services. *See* service-orientation principles, service autonomy
 with SOA, 42, 330

B

basic profile. *See* WS-I, Basic Profile

best practices
See service modeling, guidelines
See service design, guidelines

best-of-breed, 62

bottom-up approach. *See* SOA delivery strategies, bottom-up

BPEL4WS. *See* WS-BPEL

BPM, 49, 76, 387-389, 400

building block (service modeling). *See* service modeling, building block

business activities
 explained, 193-200
See also WS-BusinessActivity

business logic (SOA), 280-283

business logic (Web service). *See* SOA platforms, business logic

business process design. *See* service-oriented business process design

Business Process Execution Language for Web Services. *See* WS-BPEL

Business Process Management. *See* BPM

business service
 case study example. *See* case study examples, business service
 deriving from business models, 386-396
 explained, 127, 718
 maintenance, 480. *See also* SOA delivery strategies, agile

- See also* entity-centric business service
- See also* task-centric business service
- business service (service modeling unit).** *See* service modeling, business service
- business service layer**
 - case study example. *See* case study examples, business service layer
 - explained, 335-336, 341-344, 501, 540
- C**
- C++.** *See* .NET platform, programming languages
- C#.** *See* .NET platform, programming languages
- candidates.** *See* service candidates
- case studies**
 - background, 22-28
 - conclusion, 708-715
 - examples. *See* case study examples
 - style used to distinguish examples, 22
- case study examples**
 - abstract and concrete descriptions, 136
 - active intermediary, 122
 - addressing, 226-227
 - agile delivery strategy, 373
 - application service layer, 340
 - atomic transactions, 192
 - bottom-up delivery strategy, 369-370
 - business activity, 198-199
 - business and utility service models, 128
 - business service, 128
 - business service layer, 343
 - choosing WS-* extensions, 493-494
 - choreography and collaboration, 213-214
 - client-server architecture, 94-95
 - complex message exchange pattern, 168-169
 - complex service activity, 175-176
 - controller service model, 130
 - coordination and context management, 184-185
 - correlation, 241
 - distributed Internet architecture, 103-104
 - fire-and-forget message exchange pattern, 165-166
 - header blocks, 146
 - hybrid services, 106
 - initial sender and ultimate receiver roles, 123-124
 - intermediaries, 121-122
 - MEPs, 164-166, 168-169
 - message attachments, 148
 - message faults, 148
 - message paths, 153-154
 - message styles, 147
 - metadata exchange, 255-256
 - orchestration, 206-207
 - orchestration service layer, 345
 - passive intermediary, 121
 - policies, 247
 - positioning core standards, 488-489
 - primitive SOA, 38-39
 - publish-and-subscribe, 275-276
 - reliable messaging, 236-237
 - request-response message exchange pattern, 164

- security concepts, 265-266
- service activities, 175-176
- service compositions, 127-128
- service descriptions, 132-133, 136
- service design tools, 473
- service provider and service requestor roles, 117-119
- service registry, 141
- service roles, 117-119, 123-124
- service-orientation principle:
 - service abstraction, 299-300
- service-orientation principle:
 - service autonomy, 305-306
- service-orientation principle:
 - service composability, 302-303
- service-orientation principle:
 - service contract, 296
- service-orientation principle:
 - service discoverability, 309-310
- service-orientation principle:
 - service loose coupling, 298
- service-orientation principle:
 - service reusability, 293-294
- service-orientation principle:
 - service statelessness, 308
- service-oriented analysis
 - examples, 380-382, 388-396, 400-413, 415, 430-444
- service-oriented design examples (business process design), 588-611
- service-oriented design examples (service design), 473, 499-500, 503-521, 524-539, 542-555, 557-558, 560-562
- SOA delivery strategies 366, 369-370, 373
- SOAP, 146-148, 150
- SOAP nodes, 150
- standards, 488-489
- top-down delivery strategy, 366
- utility and business service models, 128
- vendor platforms, 680-681, 696-697
- WS-*, 493-494
- WS-Addressing language
 - examples, 616-621
- WS-BPEL language examples. *See* case study examples, service-oriented design examples (business process design)
- WS-MetadataExchange language
 - examples, 638-642
- WS-Policy language examples, 631-632, 634-635
- WS-ReliableMessaging language
 - examples, 624-628
- WS-Security language examples, 645-650
- WSDL language examples. *See* case study examples, service-oriented design examples (service design)
- choreography
 - explained, 208-215
 - See also* WS-CDL
- client-server architecture
 - administration, 93
 - and SOA. *See* SOA, compared to client-server architecture
 - application logic, 90-91
 - application processing, 91-92
 - case study example. *See* case study examples, client-server architecture
 - history, 88-90
 - security, 92-93
 - single-tier, 89
 - technology, 92
 - two tier, 90

- collaboration. *See* WS-CDL
- complex activity. *See* service activities, complex
- complex MEP. *See* MEPs, complex
- composition
 - with J2EE, 685-686
 - with .NET, 701
 - with services. *See* service-orientation principles, service composability
 - with SOA. *See* SOA, composition
- composition member. *See* composition
- concrete description. *See* WSDL, concrete description
- confidentiality. *See* security, confidentiality
- constructs. *See* specifications, constructs
- contemporary SOA
 - common characteristics, 40-54
 - concrete characteristics, 55-56
 - defined, 40, 54
 - influences, 328-329
 - maturity, 53
 - origins of concrete characteristics, 329-332
 - with J2EE. *See* J2EE platform, support for contemporary SOA characteristics
 - with .NET. *See* .NET platform, support for contemporary SOA characteristics
- context management. *See* WS-Coordination
- contract. *See* service-orientation principles, service contracts
- controller service
 - case study example. *See* case study examples, controller service model
 - explained, 128-129, 718
- coordination
 - explained, 177-185
 - See also* WS-Coordination
- coordinator service. *See* WS-Coordination, coordinator service
- correlation
 - case study example. *See* case study examples, correlation
 - explained, 238-241
 - in addressing, 240
 - in coordination, 240
 - in MEPs, 239-240
 - in orchestration, 240
 - In Plain English example, 239
 - in reliable messaging, 240-241
 - in service activities, 239-240
 - with SOA. *See* SOA, and correlation
- CORBA, 96, 100
- D**
- DCOM, 96, 100
- delivery lifecycle phases. *See* SOA, delivery lifecycle phases
- DIME, 696
- Direct Internet Message Encapsulation. *See* DIME
- DISCO. *See* .NET platform, DISCO
- discovery
 - with SOA. *See* SOA, and discovery
 - service discoverability. *See* service-orientation principles, service discoverability

distributed architecture
 administration, 102
 application logic, 98-100
 application processing, 100-101
 case study example. *See* case study examples, distributed Internet architecture
 distributed Internet, 97
 history, 95-98
 multi-tier client-server, 96
 proxy stubs, 98-99
 security, 101-102
 technology, 101
 traditional models, 65

Distributed Internet Architecture (Microsoft). *See* DNA

DNA, 698

document-style. *See* SOAP, message styles

dynamic invocation interface. *See* J2EE platform, dynamic proxy

dynamic proxy. *See* J2EE platform, dynamic proxy

E

editors. *See* service design, tools

EJBs. *See* J2EE platform, EJBs and EJB endpoints

EJB endpoints. *See* J2EE platform, EJBs and EJB endpoints

EJB Service Implementation Bean. *See* J2EE platform, Service Implementation Beans

elements. *See* specifications, elements

endpoint. *See* service endpoint

endpoint reference. *See* WS-Addressing, endpoint references

enterprise architecture, 87

Enterprise Java Beans. *See* J2EE platform, EJBs and EJB endpoints

enterprise logic, 280-283

enterprise-wide loose coupling

with J2EE, 686-687

with .NET, 702

with SOA 50, 64, 331, 334

Entity Bean. *See* J2EE platform, Entity Beans

entity business models, 364, 390-391

entity-centric business service

case study examples. *See* case study examples, service-oriented design examples (service design)

deriving candidates, 399-415, 437-442

design process. *See* service design, entity-centric business service design step-by-step process description

explained, 342, 393, 719

in service layer configurations, 347-353, 480

envelope. *See* SOAP, Envelope element

eventing

explained, 266-276

See also WS-Eventing

extensibility

extension

defined, 78-79

with J2EE, 686

with .NET, 701

with services. *See* service design, extensibility

with SOA, 48, 331

WS-*. *See* WS-*

Extensible Access Control Markup Language. *See* XACML

Extensible Markup Language. *See* XML

Extensible Rights Markup Language.
See XrML

F

federation

with J2EE, 685

with .NET, 701

with SOA, 45-46, 331

Field Guide. *See* Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services

first-generation Web services standards

See SOAP

See UDDI

See WSDL

forwarding intermediary. *See* SOAP, forwarding intermediary

G

generated stub. *See* J2EE platform, generated stub

granularity. *See* service design, interface granularity

guidelines

See service modeling, guidelines

See service design, guidelines

H

handler chain. *See* J2EE platform, service agents

handlers. *See* J2EE platform, service agents

header blocks. *See* SOAP, header blocks

headers. *See* SOAP, header blocks

HTTP Application. *See* .NET platform, service agents

HTTP Context. *See* .NET platform, service agents

HTTP handlers. *See* .NET platform, service agents

HTTP modules. *See* .NET platform, service agents

HTTP Pipeline. *See* .NET platform, service agents

HTTP Runtime. *See* .NET platform, service agents

HTTPSession object. *See* J2EE platform, HTTPSession object

HttpSessionState object, 699. *See* .NET platform, HttpSessionState object

hybrid service

case study example. *See* case study examples, hybrid services

deriving candidates, 434-436

explained, 339, 719

in service layer configurations, 347-353

problems with, 65

I

identification. *See* security, identification

immutable service contracts. *See also* service contracts, immutable

impact analysis. *See* SOA, transition plan

In Plain English examples

addressing, 221

business activity and compensation, 194

choreography, 209

coordination and context management, 178-179
 correlation, 239
 messages and message paths, 143
 message exchange patterns, 163
 metadata exchange, 249
 orchestration, 202
 publish-and-subscribe pattern, 268
 reliable messaging, 229
 security (identification, authentication, authorization), 258
 service activities, 173
 service descriptions, 131-132
 service intermediaries, 119
 service-orientation principle:
 service abstraction, 300-301
 service-orientation principle:
 service autonomy, 307
 service-orientation principle:
 service composability, 303
 service-orientation principle:
 service contract, 297
 service-orientation principle:
 service discoverability, 311
 service-orientation principle:
 service loose coupling, 298
 service-orientation principle:
 service reusability, 295
 service-orientation principle:
 service statelessness, 309
 service roles, 113-114
initial sender. *See* Web services, initial sender
integration
 considerations, 703-705
 legacy, 45-46, 61-62
 types, 704-705
 with SOA. *See* SOI
 See also interoperability

integration service. *See* application service
integrity. *See* security, integrity
interface granularity. *See* service design, interface granularity
intermediaries
 active, 121-122
 active (SOAP). *See* SOAP, active intermediary
 case study examples. *See* case study examples, intermediaries
 forwarding (SOAP). *See* SOAP, forwarding intermediary
 In Plain English example, 119
 passive, 119-120
 role, 119-120
 service intermediaries, 119-122
 See also service agents
intermediary services. *See* intermediaries
interoperability
 with J2EE, 684
 with .NET, 701
 with SOA, 45, 59-60, 330
 See also integration
intrinsic interoperability, 45

J

J2EE Connector Architecture. *See* J2EE platform, JCA
J2EE platform
 APIs, 673-674
 architecture components, 671-672
 case study example. *See* case study examples, vendor platforms
 common platform layers, 669
 development tools, 673, 684

- dynamic invocation interface, 677
- dynamic proxy, 677
- EJB containers, 669-670, 672, 675
- EJB endpoints. *See* J2EE platforms
- EJB Service Implementation Bean.
See J2EE platform, Service Implementation Beans
- EJBs and EJB endpoints, 669-672, 675-676, 678-679, 681-683
- Entity Beans, 683
- explained, 668-687
- generated stub, 677
- handler chain. *See* J2EE platform, service agents
- handlers. *See* J2EE platform, service agents
- HTTPSession object, 683
- J2EE specification, 671
- JAXB, 674, 676
- JAXM, 674
- JAXP, 670, 673
- JAXR, 670, 674, 676, 683
- JAX-RPC, 669-670, 672, 676-679, 681, 684
- JAX-RPC endpoints, 670-672, 674-676, 678-679, 681-683
- JAX-RPC Service Implementation Bean. *See* J2EE platform, Service Implementation beans
- JAX-RPC specification, 671
- JCA, 681
- JMS, 674, 682
- JSPs, 671
- packages, 673-674
- port, 675-676
- Port Component Model, 671, 675
- programming languages, 673
- relationship between layers and technologies, 669-670
- resource adapters, 685
- runtime environments, 672
- SAAJ, 674, 676
- SEI, 675
- server products, 672
- service agents, 673, 678-679
- Service Endpoint Interface. *See* J2EE platform, SEI
- Service Implementation Beans, 675-676
- service providers, 674-676
- service requestors, 677-678
- servlets, 669-670, 681
- SOAP over JMS, 682
- Stateful Session Beans, 683
- Stateless Session Beans, 670, 675-676, 683
- struts, 671
- support for contemporary SOA characteristics, 683-687
- support for primitive SOA characteristics, 680-681
- support for service-orientation principles, 682-683
- support for SOAP, 670, 673-674, 678-679, 682, 684
- support for UDDI, 670, 674, 676, 683-684
- support for WS-*. *See* J2EE platforms, vendor platform extensions
- support for WS-I Basic Profile, 684
- support for WSDL, 670, 675, 677, 684
- support for XML, 670, 673-674
- support for XSD, 670, 673-674
- support for XSLT, 673
- vendor platform extensions, 679-680, 685-687

- Web containers, 669-670, 672, 675
 - Web Services for J2EE specification, 671, 675
 - J2ME, 669
 - J2SE, 669
 - Java API for XML Messaging. *See* J2EE platform, JAXM
 - Java API for XML Processing. *See* J2EE platform, JAXP
 - Java API for XML Registries. *See* J2EE platform, JAXR
 - Java API for XML-based RPC. *See* J2EE platform, JAX-RPC
 - Java API for XML-based RPC specification. *See* J2EE platform, JAX-RPC specification
 - Java Architecture for XML Binding API. *See* J2EE platform, JAXB
 - Java Message Service API. *See* J2EE platform, JMS
 - Java Server Pages. *See* J2EE platform, JSPs
 - Java servlets. *See* J2EE platform, servlets
 - JCA. *See* J2EE platform, JCA
 - JAXB. *See* J2EE platform, JAXB
 - JAXM. *See* J2EE platform, JAXM
 - JAXP. *See* J2EE platform, JAXP
 - JAXR. *See* J2EE platform, JAXR
 - JAX-RPC. *See* J2EE platform, JAX-RPC
 - JAX-RPC endpoints. *See* J2EE platform, JAX-RPC endpoints
 - JAX-RPC Service Implementation Bean. *See* J2EE platform Service Implementation Beans
 - JAX-RPC specification. *See* J2EE platform, JAX-RPC specification
 - Java 2 Micro Edition. *See* J2ME
 - Java 2 Platform Enterprise Edition. *See* J2EE platform
 - Java 2 Platform Enterprise Edition specification. *See* J2EE platform, J2EE specification
 - Java 2 Platform Standard Edition. *See* J2SE
 - JMS. *See* J2EE platform, JMS
- L**
- legacy systems. *See* integration, legacy
 - loose coupling
 - enterprise-wide. *See* enterprise-wide loose coupling
 - with services. *See* service-orientation principles, service loose coupling
- M**
- meet-in-the-middle strategy. *See* SOA delivery strategies, agile
 - MEPs
 - and correlation. *See* correlation, in MEPs
 - and SOAP, 169
 - and WSDL. *See* WSDL, MEPs
 - case study examples. *See* case study examples, MEPs
 - complex, 166-169
 - explained, 162-172
 - fire-and-forget, 165
 - In Plain English example, 163
 - in-only pattern. *See* WSDL, in-only pattern
 - in-optional-out pattern. *See* WSDL, in-optional-out pattern
 - in-out pattern. *See* WSDL, in-out pattern

- notification. *See* WSDL, notification operation
 - one-way. *See* WSDL, one-way operation
 - out-in pattern. *See* WSDL, out-in pattern
 - out-only pattern. *See* WSDL, out-only pattern
 - out-optional-in pattern. *See* WSDL, out-optional-in pattern
 - primitive, 163-166
 - publish-and-subscribe. *See* publish-and-subscribe
 - request-response, 163-164
 - request-response (WSDL). *See* WSDL, request-response operation
 - robust in-only pattern. *See* WSDL, robust in-only pattern
 - robust out-only pattern. *See* WSDL, robust out-only pattern
 - solicit-response. *See* WSDL, solicit-response operation
 - with SOA. *See* SOA, and MEPs
 - message correlation. *See* correlation
 - message exchange patterns. *See* MEPs
 - message information headers. *See* WS-Addressing, message information headers
 - message path
 - case study example. *See* case study examples, message paths
 - explained, 152-153
 - In Plain English example, 143
 - message payload. *See* SOAP, payload
 - message reliability. *See* WS-ReliableMessaging
 - messaging. *See* SOAP
 - metadata
 - explained, 136-137
 - See also* service contracts
 - See also* service descriptions
 - See also* service endpoints
 - metadata exchange
 - explained, 248-256
 - See also* WS-MetadataExchange
 - MI headers. *See* WS-Addressing, message information headers
 - Microsoft Intermediate Language. *See* .NET platform, MSIL
 - Microsoft Messaging Queue. *See* .NET platform, MSMQ
 - modeling units. *See* service modeling, building blocks
 - MSMQ. *See* .NET platform, MSMQ
 - MTOM, 68
- N**
- namespaces
 - with SOA. *See* SOA, and namespaces
 - See also* service design, namespaces
 - .NET Framework. *See* .NET platform
 - .NET Passport, 257, 260
 - .NET platform
 - Active Server Pages. *See* .NET platform, ASP.NET
 - ASMX. *See* .NET platform, ASP.NET
 - APIs, 688, 690-692
 - architecture components, 689-690
 - ASP.NET, 688-694, 696, 698-701
 - ASP.NET Web Forms, 689
 - ASP.NET Web Services. *See* .NET platform, ASP.NET

- ASP.NET worker process, 699
- assemblies, 688-690, 692-693, 698, 700
- case study example. *See* case study examples, vendor platforms
- COM+, 691
- common platform layers, 688-689
- class library, 688, 690-692
- CLR, 688, 690-691
- common language runtime. *See* .NET platform, CLR
- development tools, 691, 700-701
- DISCO, 700
- explained, 688-702
- HTTP Application. *See* .NET platform, service agents
- HTTP Context. *See* .NET platform, service agents
- HTTP handlers. *See* .NET platform, service agents
- HTTP modules. *See* .NET platform, service agents
- HTTP Pipeline. *See* .NET platform, service agents
- HTTP Runtime. *See* .NET platform, service agents
- HttpSessionState object, 699
- Microsoft Intermediate Language. *See* .NET platform, MSIL
- Microsoft Messaging Queue. *See* .NET platform, MSMQ
- MSIL, 691
- MSMQ, 698
- programming languages, 691
- proxy class, 694-695
- relationship between layers and technologies, 688-690
- runtime environments, 691, 693
- server products, 698, 700-702
- service agents, 691-692, 695-696
- service providers, 692-693
- service requestors, 693-694
- support for contemporary SOA characteristics, 700-702
- support for primitive SOA characteristics, 697-698
- support for service-orientation principles, 698-700
- support for SOAP, 690, 692, 698
- support for UDDI, 690, 692, 699-700
- support for WS-*. *See* .NET platforms, WSE
- support for WS-I Basic Profile, 700-701
- support for WSDL, 690, 692-694, 698
- support for XML, 690-692
- support for XSD, 690-692
- support for XSLT, 690
- System.Web.Services, 690, 692
- System.Web.Services.Discovery, 692
- System.Xml, 690-692
- System.Xml.Schema, 692
- System.Xml.Xsl, 692
- UDDI SDK. *See* .NET platform, support for UDDI
- vendor platform extensions. *See* .NET platform, WSE
- Web Services Extensions. *See* .NET platform, WSE
- Windows, 688, 700
- WSE, 688-690, 695-696, 701-702
- WSE filters. *See* .NET platform, WSE

non-repudiation. *See* security, non-repudiation

notification

explained, 266-276

See also WS-Notification framework

O

OASIS

explained, 80, 82, 568

role in Web services history, 73-74

object-orientation. *See* service-orientation, compared to object-orientation

ontology, 364, 371

orchestration

explained, 200-207

See also WS-BPEL

orchestration service layer

case study example. *See* case study examples, orchestration service layer

explained, 335-336, 344-346, 395, 586

Organization for the Advancement of Structured Information Standards. *See* OASIS

organizational agility

benefit of, 63-64

explained, 51

with J2EE, 686-687

with .NET, 702

with SOA, 51, 63-64, 331, 335

P

packages. *See* J2EE platform, packages

passive intermediaries. *See* intermediaries, passive

payload. *See* SOAP, payload

performance. *See* SOA, performance considerations

policies

explained, 136, 242-248

See also WS-Policy framework

Port Component Model. *See* J2EE platform, Port Component Model

predefined process (service modeling). *See* service modeling, process activity

primitive activity. *See* service activities, primitive

primitive business activity (service modeling unit). *See* service modeling, primitive business activity

primitive business process (service modeling unit). *See* service modeling, primitive business process

primitive business service (service modeling unit). *See* service modeling, primitive business service

primitive MEP. *See* MEPs, primitive

primitive SOA

case study example. *See* case study examples, primitive SOA

components, 38

defined, 38

with J2EE. *See* J2EE platform, support for primitive SOA characteristics

with .NET. *See* .NET platform, support for primitive SOA characteristics

private registry. *See* service registries

process activity (service modeling). *See* service modeling, process activity

process service
 design. *See* service-oriented
 business process design
 explained, 203, 344-345, 395, 719
 in service layer configurations,
 347-353

proxy component. *See* service proxy

proxy service, 339-340, 368, 471

proxy stubs. *See* distributed
 architecture, proxy stubs

public registry. *See* service registries

publish-and-subscribe
 broker service, 267
 case study example. *See* case
 study examples, publish-and-
 subscribe
 concepts, 166-168, 267
 In Plain English example, 268
 publisher service, 267
 subscriber service, 267
 topics, 267
See also MEPs
See also WS-Eventing
See also WS-Notification
 framework

publisher role. *See* publish-and-
 subscribe, publisher service

Q

QoS (with SOA), 41-42, 330
 quality of service. *See* QoS

R

RailCo Ltd. *See* case studies

recommended reading. *See* Web sites,
 www.serviceoriented.ws

reliable messaging
 explained, 228-237
See also WS-ReliableMessaging

remote procedure call. *See* RPC

return on investment (with SOA).
See SOA, benefits

reusability
 with SOA, 47, 60-61, 331
 with services. *See* service-
 orientation principles, service
 reuse

ROI (with SOA). *See* SOA, benefits

RPC
 and SOAP. *See* SOAP, and RPC
 history, 73-74
 message style. *See* SOAP, message
 styles
 proxy stubs, 99
 technology, 96, 98, 102
 with SOA. *See* SOA, and RPC

S

SAAJ. *See* J2EE platform, SAAJ

SAML, 80
 assertions, 260
 authentication assertions, 260
 authorization assertions, 260
 example, 645-646
 explained, 260
 issuing authority, 260
See also single sign-on
See also security
See also WS-Security, framework

second-generation Web services
 specifications. *See* WS-*

Secure Sockets Layer. *See* SSL

security

- authentication, 259
- authorization, 259
- case study example. *See* case study examples, security concepts
- claim, 259
- concepts, 259, 261, 264
- confidentiality, 261
- digital signature. *See* XML-Signature
- encryption. *See* XML-Encryption. *See* SSL
- identification, 259
- In Plain English example, 258
- integrity, 261-262
- message-level, 262-263. *See also* WS-Security
- non-repudiation, 264
- single sign-on. *See* single sign-on token, 259
- transport-level, 262-263
See also SOA, security considerations
See also WS-Security, framework
- Security Assertion Markup Language. *See* SAML
- SEI. *See* J2EE platform, SEI
- separation of concerns. *See* service-orientation, and separation of concerns
- service abstraction. *See* service-orientation principles, service abstraction
- service abstraction layers
 - application service layer. *See* application service layer
 - business service layer. *See* business service layer

- choosing. *See* SOA composition process, choosing service layers
- configurations, 347-353, 479
- contemporary SOA characteristic, 49-52, 331
- deployment, 480
- explained, 333-353
- orchestration service layer. *See* orchestration service layer
- performance. *See* SOA, performance considerations
- standards, 479
- versioning. *See* version control
- service activities**
 - and correlation. *See* correlation, in service activities
 - and MEPs, 174
 - case study example. *See* case study examples, service activities
 - complex, 174-175
 - explained, 172-177
 - In Plain English example, 173
 - primitive, 174
 - with SOA. *See* SOA, and service activities
- service administration (delivery lifecycle phase), 361-362, 480**
- service agents, 119-121, 665-667**
- service assembly. *See* Web services, composition**
- service autonomy. *See* service-orientation principles, service autonomy**
- service candidates**
 - defined, 398-399
 - See also* service modeling

- service composability. *See* service-orientation principles, service composability
- service composition
 - case study example. *See* case study examples, service compositions
 - See also* Web services, composition
- service consumer. *See* Web services, service requestor
- service contracts
 - defined, 37, 136-137
 - immutable, 372
 - See also* metadata
 - See also* service descriptions
 - See also* service endpoints
 - See also* service-orientation principles, service contract
- service deployment (delivery lifecycle phase), 361, 365, 368, 372, 480
- service description documents. *See* service descriptions
- service design
 - application service design step-by-step process description, 451, 522-539
 - business process design. *See* service-oriented business process design
 - case study examples. *See* case study examples, service-oriented design (service design)
 - entity-centric business service design step-by-step process description, 451, 501-521
 - explained, 497-498
 - extensibility, 558
 - guidelines, 555-564
 - interface granularity, 556-557
 - metadata, 563-564
 - message styles, 146, 561-562
 - modules, 559
 - namespaces, 560
 - naming conventions, 555-556
 - prerequisites, 499
 - task-centric business service design step-by-step process description, 451, 540-555
 - tools, 471-473
 - WS-I compliance. *See* WS-I, compliance
 - See also* service-oriented design
- service descriptions
 - advertisement, 138-140
 - case study examples. *See* case study examples, service descriptions
 - explained, 131-142
 - defined, 350
 - In Plain English example, 131-132
 - See also* metadata
 - See also* service contracts
 - See also* service endpoints
- service development (delivery lifecycle phase), 360, 365, 368, 372
- service discoverability. *See* service-orientation principles, service discoverability
- Service Endpoint Interface. *See* J2EE platform, SEI
- service endpoints
 - defined, 133, 659
 - See also* metadata
 - See also* service contracts
 - See also* service descriptions

- Service Implementation Bean. *See* J2EE platform, Service Implementation Beans
- service interface layer, 282-283
- service intermediaries. *See* intermediaries
- service layer abstraction. *See* service abstraction layers
- Service Level Agreement. *See* SLA
- service loose coupling. *See* service-orientation principles, service loose coupling
- service modeling
 - building blocks, 423
 - business service, 429
 - case study examples. *See* case study examples, service-oriented analysis examples
 - explained, 398-444
 - guidelines, 416-423
 - logic classification, 423-444
 - primitive business activity, 427-428
 - primitive business service, 429
 - process activity, 428
 - step-by-step process description 399-415
 - See also* service-oriented analysis
 - See also* service-oriented business modeling
- service modeling units. *See* service modeling, building blocks
- service models
 - application service. *See* application service
 - business service. *See* business service
 - compared to service modeling building blocks, 426
 - controller service. *See* controller service
 - coordinator service. *See* WS-Coordination, coordinator service
 - entity-centric business service. *See* entity-centric business service explained, 113, 126-127
 - hybrid service. *See* hybrid service
 - integration service. *See* application service
 - process service. *See* WS-BPEL, process service
 - task-centric business service. *See* task-centric business service
 - utility service. *See* utility service
 - wrapper service. *See* wrapper service
 - See also* Appendix B, Service Models Reference
- service operation candidates. *See* service candidates
- service provider. *See* Web services, service provider
- service provider agent. *See* Web services, service provider
- service provider entity. *See* Web services, service provider entity
- service proxy, 340, 662-663, 677, 693-694
- service registries
 - and the WSDL documentation element. *See* WSDL, documentation element
 - case study example. *See* case study examples, service registry
 - explained, 139, 248
 - with SOA, 44
- service requestor. *See* Web services, service requestor

- service requestor agent. *See* Web services, service requestor
- service requestor entity. *See* Web services, service requestor entity
- service reusability. *See* service-orientation principles, service reusability
- service roles
 - case study examples. *See* case study examples, service roles
 - In Plain English example, 113-114
 - See also* Web services, roles
- service statelessness. *See* service-orientation principles, service statelessness
- service testing (delivery lifecycle phase), 360, 365, 368, 372
- service-orientation
 - and separation of concerns, 290-291
 - defined, 36
 - explained, 280-353
 - compared to object-orientation, 107-108, 291, 321-324
 - principles. *See* service-orientation principles
- service-orientation principles
 - as part of service-oriented analysis, 407
 - as part of service-oriented design, 511-512, 530-531, 549
 - and Web services, 37, 75, 324-326
 - case study examples. *See* case study examples, service-orientation principle (one example for each principle)
- compared to object-orientation principles. *See* service-orientation, compared to object-orientation
- explained, 37, 290-326
- guidelines. *See* service design, guidelines
- how principles inter-relate, 311-321
- In Plain English examples, 295, 297-298, 300-301, 303, 307, 309, 311
- service abstraction, 298-301, 316-317, 322, 325
- service autonomy, 303-307, 318-319, 323, 325
- service composability, 301-303, 317-318, 323, 325
- service contract, 295-297, 313-314, 322, 324
- service discoverability, 309-311, 320-321, 323, 325
- service loose coupling, 297-298, 315-316, 322, 324
- service reusability, 292-295, 312-313, 322, 324
- service statelessness, 307-309, 319-320, 323, 325
- with J2EE. *See* J2EE platform, support for service-orientation principles
- with .NET. *See* .NET platform, support for service-orientation principles
- See also* service-orientation
- service-oriented analysis
 - as part of delivery lifecycles, 359, 365, 372
 - case study examples. *See* case study examples, service-oriented analysis examples

- defining business automation requirements, 380
- deriving business service candidates. *See* service-oriented business modeling
- explained, 376-444
- guidelines. *See* service modeling, guidelines
- identifying existing automation systems, 380
- modeling service candidates. *See* service modeling
- objectives, 377
- parent process description, 377-382
- service-oriented architecture. *See* SOA
- Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services 6, 62, 705
- service-oriented business modeling
 - as part of service-oriented analysis, 382-396
 - with J2EE, 686
 - with .NET, 701-702
 - with SOA, 48-49, 331, 335
 - See also* service-oriented analysis
 - See also* service-oriented modeling
- service-oriented business process design
 - case study examples. *See* case study examples, service-oriented design examples (business process design)
 - explained, 392, 395, 566, 585-587
 - prerequisites, 568
 - step-by-step process description, 585-611
 - tools, 585
- service-oriented computing platform, 41
- service-oriented design
 - application service design. *See* service design, application service design step-by-step process description
 - as part of delivery lifecycles, 359, 365, 372
 - business process design. *See* service-oriented business process design
 - case study examples. *See* case study examples, service-oriented design examples (service design)
 - entity-centric business service design. *See* service design, entity-centric business service design step-by-step process description
 - explained, 448-611
 - guidelines. *See* service design, guidelines
 - objectives, 448-449
 - parent process description, 449-451
 - prerequisites, 451-452
 - SOA composition process. *See* SOA composition
 - task-centric business service design. *See* service design, task-centric business service design step-by-step process description
- service-oriented enterprise. *See* SOE
- service-oriented environment, 373
- service-oriented integration. *See* SOI
- service-oriented solutions
 - analysis. *See* service-oriented analysis

- defined, 37
- delivery lifecycle. *See* SOA, delivery lifecycle phases
- deployment. *See* service deployment (delivery lifecycle phase)
- design. *See* service-oriented design
- development. *See* service development (delivery lifecycle phase)
- managing. *See* SOA delivery strategies
- testing. *See* service testing (delivery lifecycle phase)
- services
 - agnostic, 346-347
 - analogy. *See* SOA, introductory analogy
 - communication, 35-36
 - defined, 33
 - design, 36-37. *See also* service design
 - encapsulation, 33
 - hybrid. *See* hybrid services
 - logical components, 286-290
 - relationships, 35
 - See also* service-orientation
 - See also* Web services
- servlets. *See* J2EE platform, servlets
- SGML, 72-73, 80
- SGML Open. *See* OASIS
- Simple Object Access Protocol. *See* SOAP
- single sign-on
 - explained, 257, 260-261
 - .NET Passport. *See* .NET Passport
 - SAML. *See* SAML
 - XACML. *See* XACML
- See also* security
- See also* SOA, security considerations
- SLA, 137
- SOA
 - administration. *See* service administration (delivery lifecycle phase)
 - analysis. *See* service-oriented analysis
 - and correlation, 241
 - and discovery, 44, 331, 488
 - and distributed computing, 57-58
 - and MEPs, 171
 - and namespaces, 487-488
 - and RPC, 61, 65, 77
 - and service activities, 175
 - and SOAP, 77, 486-488
 - and UDDI. *See* SOA, and discovery
 - and Web services, 56-57, 76-77
 - and WS-*, 58, 77
 - and WS-Addressing, 225-226
 - and WS-AtomicTransaction, 191
 - and WS-BPEL, 205-206, 492-493
 - and WS-BusinessActivity, 197-198
 - and WS-CDL, 212-213
 - and WS-Coordination, 183-184
 - and WS-Eventing, 274-275
 - and WS-MetadataExchange, 254-255
 - and WS-Notification framework, 274-275
 - and WS-Policy framework, 246-247
 - and WS-ReliableMessaging, 235-236
 - and WS-Security framework, 265
 - and WSDL, 485
 - and XML, 62, 67, 76-77, 482-483

- and XSD, 62, 77, 485-486
 - benefits, 59-64
 - challenges. *See* SOA, pitfalls
 - compared to client-server architecture, 88-93
 - compared to distributed Internet architecture, 95-102
 - compared to hybrid Web services architecture, 104-106
 - compared to mainframe architecture, 89
 - composition, 46-47, 61, 330
 - contemporary model. *See* contemporary SOA
 - defined, 54, 88
 - delivery lifecycle phases, 358-363
 - delivery strategies. *See* SOA delivery strategies
 - deployment. *See* service deployment (delivery lifecycle phase)
 - design. *See* service-oriented design
 - developing. *See* service development (delivery lifecycle phase)
 - ecosystem. *See* service-oriented environment
 - fundamental concepts, 32-38
 - fundamental theory, 284-290
 - governance. *See* service administration (delivery lifecycle phase)
 - history, 74-76
 - how logical components define each other 289-290
 - how logical components interrelate, 289
 - infrastructure, 63, 68-69. *See also* SOA platforms
 - introductory analogy, 32
 - logical components, 285-288
 - marketing of, 57
 - message (logical component), 286-290
 - misperceptions, 56-59
 - myths. *See* SOA, misperceptions
 - operation (logical component), 286-290
 - performance considerations, 61, 67-68, 479-480
 - pitfalls, 64-70
 - primitive model. *See* primitive SOA
 - process (logical component), 286-290
 - security considerations, 68-69, 91-92, 101. *See also* WS-Security, framework
 - service (logical component), 286-290
 - service (Web service). *See* Web services
 - skill-set requirements, 58, 61
 - standardization, 65-66
 - streamlining. *See* SOA, composition. *See also* SOA, performance considerations
 - testing. *See* service testing (delivery lifecycle phase)
 - transition plan, 66-67
 - vendors that contribute to, 82-85
See also contemporary SOA
See also primitive SOA
See also SOA platforms
- SOA composition process**
- choosing service layers, 478-480
 - explained, 450-451, 476-494
 - process description, 477-478

- SOA delivery strategies
 - agile, 370-373
 - bottom-up, 366-370
 - case study examples. *See* case study examples, SOA delivery strategies
 - explained, 362-373
 - meet-in-the-middle. *See* SOA delivery strategies, agile
 - top-down, 363-366
- SOA ecosystem. *See* service-oriented environment
- SOA platforms
 - APIs layer, 654-656
 - business logic, 658-661, 663-665
 - common SOA platform layers, 654
 - component technology layer, 654-656
 - explained, 652-705
 - fundamental software technology architecture layers, 653
 - J2EE platform. *See* J2EE platform
 - message processing logic, 658-663
 - .NET platform. *See* .NET platform
 - relationship between layers and technologies, 655-656
 - runtime layer, 654-656
 - service processing logic, 658-665
 - service provider processing tasks, 657
 - service requestor processing tasks, 657
 - Web technology layer, 654-656
- SOA Systems Inc., 19. *See also* About SOA Systems page
- SOAP
 - active intermediary, 150
 - and RPC, 99, 100, 104
 - as part of SOA platform, 656
 - attachments, 147-148
 - Body element, 143-144, 468-470
 - case study examples. *See* case study examples, SOAP
 - detail element, 470
 - Envelope element, 143-144, 467-468
 - Fault element, 148, 470
 - faultcode element, 470
 - faultstring element, 470
 - forwarding intermediary, 150
 - language, 466-471
 - header blocks, 144-145, 218
 - Header element, 143-144, 468
 - history, 73-75
 - In Plain English example, 143
 - initial sender type, 150-151
 - intermediary type, 150-151
 - message path, 152
 - message processing logic. *See* SOA platforms, message processing logic
 - message styles. *See* service design, message styles
 - mustUnderstand attribute, 468, 614
 - node types, 149-151
 - nodes, 149-154
 - receiver type, 150-151
 - payload, 144
 - payload transformation. *See* XSLT
 - payload validation. *See* XSD
 - processors, 68
 - sender type, 150-151
 - SOAP Processing Model, 150
 - ultimate receiver type, 150-151
 - with J2EE. *See* J2EE platform, support for SOAP

- with .NET. *See* .NET platform, support for SOAP
 - with SOA. *See* SOA, and SOAP
 - SOAP Message Transmission Optimization Mechanism. *See* MTOM
 - SOAP over JMS. *See* J2EE platform, SOAP over JMS
 - SOAP with Attachments API for Java. *See* J2EE platform, SAAJ
 - SOE
 - explained, 52
 - model, 424-429
 - SOI, 60, 705
 - specifications
 - constructs, 454
 - defined, 78-79
 - elements, 454
 - positioning within SOA, 480-489
 - speculative analysis. *See* SOA, transition plan
 - SSL, 68, 257, 262-263
 - standards
 - case study example. *See* case study examples, standards
 - defined, 78-79
 - design, 422-423, 479, 498
 - development, 69-70, 76, 78-85
 - naming conventions, 555-556
 - organizations, 78-85 *See also* W3C, OASIS, and WS-I
 - See also* Web services, security considerations
 - vendors that contribute to, 82-85
 - with SOA, 43, 69
 - Standard Generalized Markup Language. *See* SGML
 - Stateful Session Bean. *See* J2EE platform, Stateful Session Beans
 - stateless session beans. *See* J2EE platform, Stateless Session Beans
 - statelessness. *See* service-orientation principles, service statelessness
 - struts. *See* J2EE platform, struts
 - sub-controller service. *See* controller service
 - sub-process (service modeling). *See* service modeling, process activity
 - subscriber role. *See* publish-and-subscribe, subscriber service
- T**
- task-centric business service
 - case study examples. *See* case study examples, service-oriented design examples (service design)
 - design process. *See* service design, task-centric business service design step-by-step process description
 - deriving candidates, 399-415, 442-444
 - explained, 342, 392, 719
 - in service layer configurations, 347-353
 - TLS. *See* case studies
 - top-down strategy. *See* SOA delivery strategies, top-down
 - Transit Line Systems Inc. *See* case studies
 - transition architectures. *See* SOA, transition plan
 - transition plan. *See* SOA, transition plan
 - transformation. *See* XSLT

U**UDDI**

- and metadata exchange, 253
- as part of SOA platform, 656
- binding template, 140
- business entity, 140
- business service, 140
- explained, 44, 75, 102, 111, 139-141
- tModel, 140
- with J2EE. *See* J2EE platform, support for UDDI
- with .NET. *See* .NET platform, support for UDDI
- with SOA. *See* SOA, and discovery

UDDI.org, 74

ultimate receiver. *See* Web services, ultimate receiver

Universal Description, Discovery, and Integration. *See* UDDI

utility application service. *See* utility service

utility service

- case study example. *See* case study examples, utility and business service models
- explained, 127, 719
- See also* application service

V

validation. *See* XSD

vendor diversity

- with J2EE, 684
- with .NET, 700
- with SOA, 44, 63, 330

vendor platforms

- J2EE. *See* J2EE platform
- .NET. *See* .NET platform

version control

- with SOA, 480
- See also* immutable service contracts
- See also* WS-MetadataExchange

Visual Basic. *See* .NET platform, programming languages

W**W3C**

- explained, 79-80, 82
- role in Web services history, 73-74

Web container. *See* J2EE platform, Web containers

Web services

- activity, 175
- business logic. *See* SOA platforms, business logic
- composition, 124-125, 479-480. *See also* service abstraction layers
- concepts, 112-130
- designing for SOA. *See* service-oriented design
- extensions. *See* Web services, specifications
- framework, 111-154
- granularity. *See* service design, interface granularity
- history, 73-74
- initial sender, 122-123
- logical components, 284-285
- message processing logic. *See* SOA platforms, message processing logic
- naming. *See* service design, naming conventions
- non-SOA architecture, 104-106

- performance considerations. *See* SOA, performance considerations
- physical architecture. *See* SOA platforms
- processing logic. *See* SOA platforms, service processing tasks
- roles, 113-123
- security considerations. *See* SOA, security considerations
- service agents. *See* service agents
- service consumer. *See* Web services, service requestor
- service provider, 114-115, 657-662, 664-665
- service provider agent. *See* Web services, service provider
- service provider entity, 115
- service requestor, 116-117, 657, 660-665
- service requestor agent. *See* Web services, service requestor
- service requestor entity, 115
- specifications. *See* first-generation Web services standards. *See also* WS-*
- third-party marketplace, 74
- underlying logic. *See* SOA platforms, business logic
- ultimate receiver, 122-123
- with SOA. *See* SOA
- Web Services Business Process Execution Language.** *See* WS-BPEL
- Web Services Choreography Description Language.** *See* WS-CDL
- Web Services Description Language.** *See* WSDL
- Web Services Extensions.** *See* .NET platform, WSE
- Web Services Flow Language.** *See* WSFL
- Web Services for J2EE specification.** *See* J2EE platform, Web Services for J2EE
- Web Services Interoperability Organization.** *See* WS-I
- Web sites**
 - www.oasis-open.org, 568
 - www.serviceoriented.ws, 7, 19, 20, 457, 521, 539, 555, 600, 652, 668, 688
 - www.soaplanning.com. *See* About SOA Systems page
 - www.soasystems.com, 19. *See* About SOA Systems page
 - www.soatraining.com. *See* About SOA Systems page
 - www.specifications.ws, 157, 257, 274, 680, 696
 - www.thomaserl.com, 20
 - www.w3c.org, 68
 - www.ws-i.org, 563
 - www.ws-standards.com, 141
 - www.xwif.com, 19
 - www.xmltechnologyexpert.com, 7
- Windows.** *See* .NET platform, Windows
- World Wide Web Consortium.** *See* W3C wrapper service, 104-105, 339, 719
- WS-***
 - as part of SOA platform, 656
 - case study example. *See* case study examples, WS-*
 - choosing extensions, 490-494
 - concepts, 155-276
 - defined, 76, 157

- how specifications inter-relate, 161, 219
 - with J2EE. *See* J2EE platform, support for WS-*
 - with .NET. *See* .NET platform, WSE
- WS-Addressing
 - action, 225
 - Action element, 618
 - address, 223
 - Address element (endpoint reference), 616
 - and correlation. *See* correlation, in addressing
 - case study examples. *See* case study examples, addressing. *See also* case study examples, WS-Addressing language examples
 - concepts, 200-228
 - destination, 224
 - endpoint references, 222-223
 - EndpointReference element, 616-617
 - explained, 200-228, 615-622
 - fault endpoint, 224
 - FaultTo element, 618
 - From element, 618
 - In Plain English example, 221
 - language, 615-622
 - message id, 224
 - message information header elements, 617-620
 - message information headers, 223-225
 - MessageID element, 617
 - policy, 223
 - Policy element (endpoint reference), 616
 - PortName element (endpoint reference), 616
 - PortType element (endpoint reference), 616
 - reference parameters, 223
 - reference properties, 223
 - ReferenceParameters element (endpoint reference), 616
 - ReferenceProperties element (endpoint reference), 616
 - RelatesTo element, 617
 - ReplyTo element, 617
 - relationship, 225
 - relationship to other specifications, 615
 - reply endpoint, 224
 - service port type, 223
 - ServiceName element (endpoint reference), 616
 - source endpoint, 224
 - To element, 618
 - with SOA. *See* SOA, and WS-Addressing
- WS-AtomicTransaction
 - and WS-Coordination, 180
 - atomic transaction coordinator model, 188-189
 - case study example. *See* case study examples, atomic transactions
 - commit phase, 189-190
 - Completion protocol, 188
 - concepts, 186-193
 - Durable 2PC protocol, 188
 - prepare phase, 189-190
 - protocols, 188
 - Volatile 2PC protocol, 188
 - with SOA. *See* SOA, and WS-AtomicTransaction

- WS-Attachments, 696
- WS-BaseFaults. *See* WS-Resource framework
- WS-BaseNotification. *See* WS-Notification framework
- WS-BPEL, 101
 - and coordination, 205
 - and correlation. *See* correlation, in orchestration
 - and service activities, 205
 - and XSD, 571-572
 - assign element, 577-578
 - basic activities, 204
 - business protocols, 203
 - case element, 577
 - case study example. *See* case study examples, orchestration. *See also* case study examples, service-oriented design examples (business process design)
 - catch element, 578
 - catchAll element, 578
 - compensationHandler element, 579
 - concepts, 76, 203-204
 - condition attribute, 577
 - copy element, 577-578
 - correlationSets element, 240, 579
 - createInstance attribute (receive), 575
 - else element, 577
 - elseif element, 577
 - empty element, 579
 - eventHandlers element, 579
 - exit element, 579
 - explained, 76, 203-204, 566-580
 - faultHandlers element, 578
 - flow, 204
 - flow element, 589
 - from element, 577-578
 - getVariableData function, 572-573
 - getVariableProperty function, 572-573
 - history, 567-568
 - if element, 577
 - In Plain English example, 202
 - inputVariable attribute (invoke), 574
 - invoke element, 574
 - language, 566-580
 - link, 204
 - messageExchange attribute, 576
 - messageType attribute, 571-572
 - myRole attribute, 569-570
 - otherwise element, 577
 - operation attribute (invoke), 574
 - operation attribute (receive), 575
 - operation attribute (reply), 576
 - outputVariable attribute (invoke), 574
 - part attribute, 578
 - partner links, 203
 - partner services, 203
 - partnerLink attribute (invoke), 574
 - partnerLink attribute (receive), 575
 - partnerLink attribute (reply), 576
 - partnerLink element, 569-570
 - partnerLinks element, 569-570
 - partnerLinkType element, 570-571
 - partnerRole attribute, 569-570
 - pick element, 579
 - portType attribute (invoke), 574
 - portType attribute (receive), 575
 - portType attribute (reply), 576
 - process definition, 203

- process element, 568-569
- process service. *See* process service
- query attribute, 578
- receive element, 575
- reply element, 576
- role element, 571
- scope element, 580
- sequence, 204
- sequence element, 573
- structured activities, 204
- switch element, 577
- synchronization dependencies, 204
- terminate element, 580
- throw element, 580
- to element, 577-578
- tools. *See* service-oriented business process design, tools
- variable attribute (receive), 575
- variable attribute (reply), 576
- variables element, 571-572
- wait element, 580
- while element, 580
- with SOA. *See* SOA, and WS-BPEL
- WS-BrokeredNotification.** *See* WS-Notification framework
- WS-BusinessActivity**
 - and atomic transactions, 196-197
 - and WS-Coordination, 180
 - business activity coordinator model, 195
 - BusinessAgreementWithCoordinatorCompletion protocol, 195
 - BusinessAgreementWithParticipantCompletion protocol, 194
 - cancellation notifications, 196
 - cancelled state, 196
 - case study example. *See* case study examples, business activity
 - compensation process, 193-194
 - compensation state, 196
 - completed notification, 196
 - completed state, 196
 - concepts, 193-200
 - exit notification, 196
 - In Plain English example, 194
 - protocols, 194-195
 - states, 195-196
 - with SOA. *See* SOA, and WS-BusinessActivity
- WS-CDL 80,**
 - and orchestrations, 211-212
 - case study example. *See* case study examples, choreography and collaboration
 - channels, 210
 - choreography composition, 211
 - collaboration, 209-210
 - concepts, 208-215
 - In Plain English example, 209
 - interactions, 210
 - modules, 210-211
 - participants, 210
 - relationships, 210
 - roles, 210
 - with SOA. *See* SOA, and WS-CDL
 - work units, 210
- WS-Coordination, 101**
 - activation service, 179-183
 - and correlation. *See* correlation, in coordination
 - case study example. *See* case study examples, coordination and context management
 - completion acknowledgement message, 182
 - completion request message, 182

- concepts, 177-185
 - coordination context, 180-181
 - coordination protocols, 180
 - coordination type, 180
 - CoordinationContext element, 581-582
 - CoordinationType element, 582-584
 - coordinator service, 179-180, 719
 - CreateCoordinationContext message, 181
 - Expires element, 583
 - explained, 177-185, 581-584
 - Identifier element, 583
 - In Plain English example, 178-179
 - language, 581-584
 - participant, 180-181
 - protocol-specific services, 179-180
 - registration service, 179-183
 - RegistrationService element, 583
 - ReturnContext message, 181
 - with SOA. *See* SOA, and WS-Coordination
 - WS-AtomicTransaction coordination type, 584
 - WS-BusinessActivity coordination type, 584
- WS-Eventing, 167**
 - concepts, 266-276
 - event sink, 272
 - event source, 272
 - example, 621
 - notification message, 272
 - subscribers, 272
 - subscription end message, 272
 - subscription filter, 273
 - subscription managers, 272
 - subscription messages, 272-273
 - with SOA. *See* SOA, and WS-Eventing
- WS-Federation, 257**
- WS-I**
 - Attachments, 680
 - Basic Profile, 81, 111, 483-485
 - Basic Security Profile, 257
 - compliance, 473, 559-560, 562-563
 - conformance claims, 564
 - explained, 81-82
 - with J2EE. *See* J2EE platform, support for WS-I Basic Profile
 - with .NET. *See* .NET platform, support for WS-I Basic Profile
- WS-MetadataExchange**
 - and discovery, 252-253
 - and version control, 253-254
 - case study example. *See* case study examples, metadata exchange. *See also* case study examples, WS-MetadataExchange language examples
 - concepts, 248-256
 - Dialect element, 638-641
 - explained, 248-256, 636-642
 - Get Metadata request, 250-251, 637-638
 - Get Metadata response, 250-251
 - Get request, 251-252, 641-642
 - Get response, 251-252
 - GetMetadata element, 637-638
 - Identifier element, 639-641
 - In Plain English example, 249
 - language, 636-642
 - Metadata element, 640-641
 - MetadataReference element, 640-641

- MetadataSection element, 640-641
 - relationship to other specifications, 637
 - with SOA. *See* SOA, and WS-MetadataExchange
- WS-Notification framework, 167
 - concepts, 266-276
 - example, 620-621
 - notification broker, 270-271
 - notification consumer, 269
 - notification messages, 269
 - notification producer, 269
 - publisher registration manager, 271
 - situation, 269
 - subscription manager, 271
 - topics, 269
 - with SOA. *See* SOA, and WS-Notification framework
- WS-Policy framework
 - All element, 632
 - case study example. *See* case study examples, policies. *See also* case study examples, WS-Policy language examples
 - concepts, 243-248
 - ExactlyOne element, 631-632
 - explained, 243-248, 629-636
 - Ignored attribute (usage), 633
 - in choreography, 246
 - in coordination, 246
 - in orchestration, 246
 - in reliable messaging, 246
 - language, 629-636
 - Language element, 630
 - MessagePredicate element, 631
 - Observed attribute (usage), 633
 - OneOrMore element, 632
 - Optional attribute (usage), 633
 - policy alternative vocabulary, 245
 - policy alternatives, 244-245
 - policy assertion types, 245
 - policy assertions, 244-245, 635-636
 - policy attachments, 245
 - Policy element, 630-631
 - policy expressions, 245
 - policy scope, 245
 - policy subject, 245
 - policy vocabularies, 245
 - PolicyAttachment element, 635
 - PolicyReference element, 633-634
 - PolicyURIs attribute, 634
 - Preference attribute, 631-633
 - Rejected attribute (usage), 633
 - relationship to other specifications, 630
 - Required attribute (usage), 633
 - SpecVersion element, 631
 - TextEncoding element, 630
 - URI attribute, 633
 - Usage attribute, 631-633
 - with SOA. *See* SOA, and WS-Policy framework
- WS-PolicyAssertions. *See* WS-Policy framework
- WS-PolicyAttachments. *See* WS-Policy framework
- WS-Referral, 696
- WS-ReliableMessaging
 - AcknowledgementInterval, 628
 - AcknowledgementRange element, 625-626
 - acknowledgements, 231-232
 - AckRequested element, 627-628
 - and addressing, 235

- and correlation. *See* correlation, in reliable messaging
 - application destination, 230
 - application source, 230
 - AtLeastOnce delivery assurance, 233-234
 - AtMostOnce delivery assurance, 233
 - BaseRetransmissionInterval element, 628
 - case study example. *See* case study examples, reliable messaging. *See also* case study examples, WS-ReliableMessaging language examples
 - concepts, 228-237
 - delivery, 230
 - delivery assurances, 233-235
 - ExactlyOnce delivery assurance, 233-234
 - Expires element, 628
 - history, 84
 - Identifier element, 624-625
 - In Plain English example, 229
 - InactivityTimeout element, 628
 - InOrder delivery assurance, 235
 - language, 622-629
 - last message identifier, 230
 - LastMessage element, 623-624
 - Lower attribute, 625
 - message number, 230
 - MessageNumber element, 623-626
 - Nack element, 626-627
 - negative acknowledgements, 231
 - relationship to other specifications, 623
 - request acknowledgements, 231
 - RM destination, 230
 - RM source, 230
 - sequence, 230
 - sequence acknowledgement, 231-232
 - Sequence element, 623-624
 - SequenceAcknowledgement element, 625-626
 - SequenceCreation element, 628
 - SequenceRef element, 628
 - transmission, 230
 - Upper attribute, 625
 - with SOA. *See* SOA, and WS-ReliableMessaging
- WS-Reliability, 84**
- WS-Resource framework, 680**
- WS-ResourceLifetime. *See* WS-Resource framework**
- WS-ResourceProperties. *See* WS-Resource framework**
- WS-SecureConversation, 257**
- WS-Security**
- actor attributes, 644
 - BinarySecurityToken element, 644
 - case study examples. *See* case study examples, security concepts. *See also* case study examples, WS-Security language examples
 - concepts. *See* security explained, 68, 80, 101, 642-650
 - framework, 68, 80, 101
 - language, 642-650
 - Password element, 644-645
 - relationship to other specifications, 643
 - Security element, 644-646
 - SecurityTokenReference element, 644

- specifications, 257
- Username element, 644-645
- UsernameToken element, 644-645
- with SOA. *See* SOA, and
 - WS-Security framework
- XML-Encryption. *See* XML-Encryption
- XML-Signature. *See* XML-Signature
- See also* security
- See also* single sign-on
- See also* SOA, security considerations
- WS-SecurityPolicy, 257, 265
- WS-ServiceGroup. *See* WS-Resource framework
- WS-Topics. *See* WS-Notification framework
- WSDL
 - abstract description, 134-135, 457-458
 - as part of SOA platform, 656
 - auto-generation. *See* service design, tools. *See also* service proxy
 - binding element, 135-136, 463-465, 469
 - case study examples. *See* case study examples, service-oriented design examples (service design)
 - concrete description, 134-136, 457-458
 - editors. *See* service design, tools
 - explained, 131-137
 - definitions element, 458-459
 - documentation element, 466
 - endpoint element, 136, 465
 - granularity. *See* service design, interface granularity
 - history, 73, 75, 80
 - import, 465-466, 559-560
 - in-only pattern, 170
 - in-optional-out pattern, 171
 - in-out pattern, 170
 - input element, 135, 462-465
 - interface element, 135, 462
 - language, 457-466
 - MEPs, 169-171, 463
 - message attribute, 462-463
 - message element, 135, 460-462, 469
 - modules. *See* service design, modules
 - name attribute, 461
 - namespaces. *See* service design, namespaces
 - notification operation, 169-170
 - one-way operation, 169-170
 - operation element, 135, 462-464
 - out-in pattern, 170
 - out-only pattern, 171
 - out-optional-in pattern, 171
 - output element, 135, 462-465
 - processing of. *See* SOA platforms, service processing tasks part, 461-462
 - port element, 135-136, 465
 - portType element, 135, 462-463, 570-571
 - request-response operation, 169-170
 - robust in-only pattern, 171
 - robust out-only pattern, 171
 - service element, 135-136, 465
 - soap: qualifier, 459, 463-464

- solicit-response operation, 169-170
- style attribute, 464-465, 469, 561-562
- type attribute, 461
- types element, 453-454, 459-461, 469
- use attribute, 465, 469, 561-562
- version control. *See* version control
- with J2EE. *See* J2EE platform, support for WSDL
- with .NET. *See* .NET platform, support for WSDL
- with SOA. *See* SOA, and WSDL
- xmlns attribute, 459
- XSD elements related to WSDL. *See* XSD, WSDL-related XSD schema elements
- WSDL definition. *See* WSDL, explained
- WSE. *See* .NET platform, WSE
- WSE filters. *See* .NET platform, WSE
- WSFL, 568
- WSRM. *See* WS-ReliableMessaging
- X**
- XACML, 80, 257, 260
- XKMS, 257
- XLANG, 568
- XML
 - architecture, 67
 - benefits, 62
 - challenges, 62
 - history, 72-73, 79
 - tutorials. *See* Web Sites, www.xmltechnologyexpert.com
 - with J2EE. *See* J2EE platform, support for XML
 - with .NET. *See* .NET platform, support for XML
 - with SOA. *See* SOA, and XML
 - XML Key Management. *See* XKMS
 - XML Schema. *See* XSD
 - XML Schema Definition Language. *See* XSD
 - XML & Web Services Integration Framework. *See* XWIF
 - XML-binary Optimized Packaging. *See* XOP
 - XML-Encryption, 257
 - CipherData element, 647-648
 - CipherReference element, 647-648
 - CipherValue element, 647-648
 - concepts, 261, 263-264
 - EncryptedData element, 646-647
 - keys, 264
 - Type attribute, 647
 - See also* WS-Security, framework
 - XML-RPC, 74
 - XML-Signature, 257
 - CanonicalizationMethod element, 648, 650
 - concepts, 261-264
 - DigestMethod element, 648, 650
 - DigestValue element, 648, 650
 - KeyInfo element, 649-650
 - keys, 264
 - non-repudiation, 264
 - Reference element, 649-650
 - Signature element, 649-650
 - SignatureMethod element, 649-650
 - SignatureValue element, 649-650
 - SignedInfo element, 649-650
 - See also* WS-Security, framework

XMLTC Consulting Inc. *See* SOA Systems Inc.

XOP, 68

XrML, 257

XSD, 137-138

- benefits, 62
- complexType element, 455-456, 459-460
- design process. *See* service design
- editors. *See* service design, tools
- element element, 455
- elementFormDefault attribute, 455
- history, 73, 79
- import element, 456
- include element, 456
- language, 453-457
- namespaces. *See* service design, namespaces
- other important XSD elements and attributes, 456-457
- schema element, 454-455, 460
- sequence element, 456
- simpleType element, 455-456
- targetNamespace attribute, 455, 560
- with J2EE. *See* J2EE platform, support for XSD
- with .NET. *See* .NET platform, support for XSD
- with SOA. *See* SOA, and XSD
- with WS-BPEL. *See* WS-BPEL, and XSD
- WSDL-related XSD schema elements, 453-457

XSL Transformations. *See* XSLT

XSLT, 68, 73, 79, 658-660

See also J2EE platform, support for XSLT

See also .NET platform, support for XSLT

XWIF, 19