

# Index

## A

- ABAC (Attribute-Based Access Control), 397, 399
- abstract classes, in derivation, 192–193
- abstract processes
  - process modeling with, 303
  - turning executable processes into, 305–306, 308
  - using BPEL with, 300, 308–309
- abstraction
  - as architectural principle, 31
  - avoiding SOA stovepipes by using, 199–200
  - evaluating Design Characteristics for, 593, 606–607
  - overview of, 163–164
- abstraction layer
  - CICS integration using, 558–559
  - integration service using, 360–361
  - as service bus, 345
- access control
  - authentication, 392–395
  - authorization, 395–396
  - case study example, 441–442
  - DAC and MAC, 396–397
  - for data, not just services, 427–428
  - defined, 399
  - enforcing policy. *See* policy enforcement approaches
  - federated identity and cross-enterprise, 397–400
  - security gateway rules, 331
- Access Control Policy Management Service, 418
- AccessInsuredLocation operation, 547, 552–553
- accommodation of change, 31
- ACID (atomicity, consistency, isolation, and durability) transactions, 295, 382–383
- ACME Insurance
  - domains, 153
  - service implementation design, 268–272
- ACME Insurance, service interface design
  - conceptual architecture, 225–227
  - overview of, 224–225
  - problem space model. *See* problem space model
  - solution model, 241–249
  - use cases. *See* business use cases
- ACME Insurance, service-based integration, 541–578
  - AccessInsuredLocation, 552–553
  - ACME Insurance, 542–547
  - based on existing Java APIs, 568–570

- ACME Insurance, service-based integration (*continued*)
  - based on vendor's Web Services, 576–578
  - CollectPolicyFinancials, 555–556
  - establishing policy submission, 548–551
  - with existing CICS transactions, 557–565
  - with existing COM components, 565–568
  - with existing databases, 573–576
  - with existing J2EE applications, 570–573
  - GeneratePolicyNoticeDocuments, 554–555
  - GetDriversInformation, 553–554
  - high-level integration design, 547–548
  - rate insurance policy, 551–552
  - requirements for ACME implementation, 556–557
  - work with documents, 556
- ACORD (The Agency Company Organization for Research and Development)
  - Create Quote operation, 271
  - document receipt and, 261
  - overview of, 246
- actions, activity diagram, 257–258
- activities
  - with conditional outputs, 148–149
  - islands of automation focusing on limited set of, 355
  - middle-out process design, 110
  - service composition using human, 297–298
  - value chain, 125–126, 499–500
- activity diagrams, 257–258
- actors, 126, 228–229
- administration, ESB, 347
- The Agency Company Organization for Research and Development. *See* ACORD (The Agency Company Organization for Research and Development)
- agility, 16–18
- alignment
  - Business Motivation Model, 136–137
  - overview of, 17–18
  - of services with business, 40–41
- Alignment Characteristics, evaluating SOA, 597–605
  - business alignment, 598–599
  - defined, 597–598
  - fit for purpose, 601–602
  - security, 602–603
  - Semantic Alignment, 603–605
  - service evaluation matrix overview, 591–592
  - specification, 599–601
- #all keyword, 195
- Anonymous SSO, 413
- APIs (application programming interfaces)
  - building SOA stovepipes, 100
  - integration based on existing Java, 557, 568–570
  - logger, 335–336
  - transforming into new service interfaces, 16
- application architecture
  - 3-tiered, 317–321
  - Enterprise Architecture, 45–46
  - n*-tiered, 318–321
  - SOA enterprise solutions vs., 312–313
- application developers, 461
- application exceptions, service interface design, 220
- application metamodel, 74–75
- application profile, 74–75
- application programming interfaces. *See* APIs (application programming interfaces)
- application servers, 421–422
- applications
  - agility, flexibility and alignment architecture, 17
  - insurance case study of existing IT, 543–545
  - integration requirements, 15–16
  - SOA enterprise solutions vs., 312–313

- architectural models
  - overview of, 74–75
  - service composition, 279–281
- architectural views, 30–31
- architecture. *See also* SOA (Service-Oriented Architecture), promise of
  - avoiding death by, 32–33
  - business. *See* BA (business architecture)
  - case study of overall integration, 547–548
  - complete analysis for security, 437
  - conceptual, 226–227, 497–498
  - defining, 28–29
  - enterprise solution, 313–317
  - life cycle process, 83
  - principles and practices of, 30–32
  - styles, 29–30
- architecture fundamentals, 33–50
  - architecture defined, 28–29
  - business-driven, 41–44
  - common semantics and data, 40
  - determining how to use, 41
  - elements of, 33–38
  - Enterprise Architecture vs., 44–46, 49–50
  - integrating packaged and legacy systems, 39
  - principles, 30–33
  - reference architecture, 73–75
  - services. *See* services
  - software architecture, 46–50
  - styles, 29–30
  - technology infrastructure, 39–40
- artifacts
  - dependency management, 484
  - finding with service repository, 483–485
  - middle-out process design, 111
  - Reusable Asset Specification, 485–486
  - service, 38
- aspect partitioning, 152
- assertions, SAML, 411–412, 435
- Assessments, Business Motivation Model, 133
- associations
  - defined in information model, 166
  - modeling using specializations, 172–174
  - multiplicity of, 166–167
- atomic service, 62
- atomicity, consistency, isolation, and durability (ACID) transactions, 295, 382–383
- attesting trust, 422–423
- Attribute Service, 418, 440
- Attribute SSO, 413
- attribute statements, SAML assertions, 412
- Attribute-Based Access Control (ABAC), 397, 399
- attributes
  - decentralized policy management, 429–430
  - derived, 174–175
  - identifiers as, 163–167
  - information model, 163–167
  - modeling using specializations and, 172–174
- auditing
  - developing enterprise policy, 478
  - security and, 355–356
  - troubleshooting and, 435–436
- Auditing Services, 418
- authentication
  - best practices, 419
  - case study example, 437–442
  - defined, 400
  - of enterprise applications, 355–357
  - identity propagation for SSO, 420–425
  - overview of, 392–395
  - point-to-point, 425–426
  - SAML assertions, 412
  - SSL. *See* SSL authentication
  - SSL/TLS protocols, 393, 400–403
  - travel insurance case study, 519–521
- authorization. *See also* access control
  - for access control, 395–396
  - defined, 395
  - enforcing DAC and MAC, 396–397

authorization. *See also* access control  
 (continued)  
 enterprise applications, 355–357  
 overview of, 400  
 travel insurance case study, 521–522  
 Authorization Decision Service, 418,  
 440  
 authorization decision statements,  
 SAML, 412  
 automation, islands of, 355  
 Automobile LOB Pricing service,  
 269–271  
 autonomy, service  
 dealing with service changes, 326  
 evaluating SOA, 596, 618–619  
 overview of, 54  
 SCA references and, 284

**B**

B2B interactions, 206, 313–317  
 BA (business architecture). *See also*  
 BMM (Business Motivation Model)  
 basing SOA project on, 114  
 creating semantic information model,  
 88–90  
 defining, 121–122  
 designing, 121–124  
 enterprise, 124  
 Enterprise Architecture and, 44, 46  
 getting started, 85–86  
 implementing, 78  
 methodology, 80–81, 85–86  
 project, 124–125  
 service and information models,  
 127–130  
 service design process, 502–505  
 SOA driven by, 41–44  
 technology of, 130–132  
 BAM (business activity monitoring),  
 338–340  
 banking, customer service solution,  
 4–7  
 behaviors, modeling, 172–174  
 binding, dynamic, 54–55, 56  
 biometrics, and authentication, 393  
 BMM (Business Motivation Model),  
 132–137

alignment and traceability, 136–137  
 defining, 131  
 Ends, 134  
 Influencers, 136  
 Means, 134–136  
 overview of, 132–133  
 service design process, 503  
 travel insurance case study, 501–502  
 bottom-up SOA, service design,  
 108–113  
 BPDM (Business Process Definition  
 Metamodel), 132  
 BPEL (Business Process Execution  
 Management), 42, 140  
 BPM (Business Process Management).  
*See also* Business Process Models  
 benefits, limitations of, 141–142  
 creating reference architecture using,  
 85  
 defining, 64  
 differentiating service policies with,  
 463  
 engines, 286, 294–296  
 implementing services and SOA with,  
 42–43  
 marketing hype or reality of, 141–142  
 modeling and, 137–138  
 purposes of, 70–72  
 server, centralized orchestration,  
 290–292  
 service composition and, 278–279  
 BPMN (Business Process Modeling  
 Notation)  
 defined, 131–132  
 example of, 138–139  
 executable models and, 140–141  
 problem model scenario diagrams  
 using, 229  
 service composition and, 278–279  
 BRM (Business Rules Management),  
 70–72  
 Browser-Based SSO  
 federated identity and, 398  
 identity propagation with REST  
 using, 424–425  
 using trusted token service, 423–424

- BU01- quote insurance (business use case)
  - actors, 228
  - alternative workflow, 582
  - basic workflow, 580–582
  - detailed scenario diagrams, 234–238
  - document model, 248–249
  - information model, 246–248
  - initial scenario diagrams, 229–232
  - overview of, 579
  - performance goals, 582
  - problem space model, 227–228
- BU02- process application (business use case)
  - actors, 228
  - alternative workflow, 585
  - basic workflow, 583–584
  - detailed scenario diagrams, 234–238
  - document model, 248–249
  - information model, 246–248
  - initial scenario diagrams, 229–232
  - overview of, 582
  - performance goals, 585
  - problem space model, 227–228
- BU03- change policy (business use case)
  - actors, 228
  - alternative workflow, 587–588
  - basic workflow, 585–587
  - detailed scenario diagrams, 234–238
  - document model, 248–249
  - extension points, 588
  - information model, 246–248
  - initial scenario diagrams, 229–232
  - overview of, 585
  - problem space model, 227–228
  - use cases, business, 585–588
- BU04- cancel insurance (business use case)
  - actors, 228
  - detailed scenario diagrams, 234–238
  - document model, 248–249
  - information model, 246–248
  - initial scenario diagrams, 229–232
- bus, ESB, 346
- business, starting with, 119–157. *See also* BA (business architecture)
  - business architecture, 121–124
  - business architecture technology, 130–132
  - business context, 126–130
  - Business Motivation Model, 132–137
  - Business Process Management, 137–138
  - Business Process Models. *See* Business Process Models
  - enterprise business architecture, 124
  - organizing services, 151–155
  - overview of, 119–120
  - project business architecture, 124–125
  - service inventory, 155–156
  - travel insurance case study, 498–502, 506–508
  - value chain, 125–126
- business activity monitoring (BAM), 338–340
- business alignment, 40–41, 591, 598–599
- business architecture. *See* BA (business architecture)
- business components
  - implementing data mapping, 378–379
  - implementing database-based integration, 574–576
  - implementing integration with direct database access, 376
  - implementing integration with J2C, 373
  - implementing MOM-based integration, 366
  - integration access with, 361–362
  - integration based on vendor’s Web Services, 577–578
  - layer, 363–364
- business context diagram
  - defined, 122
  - overview of, 126–130
  - service design process, 503
  - travel insurance case study, 506–508
- business derivation, 31

- business domain-independent
  - interoperability, 162
- business domain-specific
  - interoperability, 161–162
- business layer. *See* service business layer
- business logic, 264–267
- business metamodel, reference
  - architecture, 73, 75
- business model, 78, 83–85, 222–224
- Business Motivation Model. *See* BMM (Business Motivation Model)
- Business Perspective, 590–597
- business process architect, 460
- business process controller, 315–317, 341
- Business Process Definition Metamodel (BPDM), 132
- Business Process Execution Management (BPEL), 42, 140
- Business Process Management. *See* BPM (Business Process Management)
- Business Process Modeling Notation. *See* BPMN (Business Process Modeling Notation)
- Business Process Models, 138–155
  - analysis and design, 24–25
  - Business Process Management vs., 138–139
  - Business Process Modeling Notation, 131–132
  - components, 139–140
  - conditional flows, 148
  - conditional operation outputs, 148–149
  - creating, 143–148
  - defined, 36, 64
  - executable, 140–141
  - overview of, 22
  - processes and services, 149–150
  - service design using, 107–108
  - in SOA world, 142–143
  - travel insurance case study, 509–510
- business processes
  - business rules vs., 292–295
  - defined, 62, 86
  - describing business model scenario, 223–224
  - diagram, 122
  - getting started, 86–88
  - implementing, 79
  - implementing with business rules, 294
  - integration of, 357
  - islands of automation duplicating, 355
  - methodology, 86–88
  - overview of, 36
  - service design process, 504
  - service granularity for, 56
  - service policies vs., 463
  - travel insurance case study, 509–510
- business profile, reference architecture, 74–75
- business rules
  - Business Motivation Model, 502
  - Business Rules Management, 70–72
  - context-based routing and, 366
  - directives governing, 135–136
  - service composition and, 292–295
- Business Rules Group, 132
- Business Rules Management (BRM), 70–72
- business services
  - Business Process Models defining, 64
  - characteristics of, 207
  - common patterns of, 68–70
  - creating business processes from, 120
  - defined, 62
  - as foundation for business processes, 315
  - granularity of, 57
  - implementing business processes, 87–88
  - implementing with business rules, 294
  - integration of, 360–364
  - integration services vs., 364
  - interface design requirements, 216–217
  - organizing service inventory, 232–233

- overview of, 35–36
  - service hierarchy, 58–59, 275–276
  - specifications, 95–96
  - types and purpose of, 70–72
  - typical interface combinations, 217
  - business use cases, 579–588
    - BU01- quote insurance, 579–582
    - BU02- process application, 582–585
    - BU03- change policy, 585–588
  - business value chain. *See* value chain diagram
  - business workers, 228
  - business-domain semantics, 201
- C**
- CalculatePolicyFinancials operation, 546
  - callable agent, 343
  - CancelInsurancePolicy operation, 545, 550–551, 556
  - canonical data format, Message Brokers, 368–369
  - case studies
    - service-based integration. *See* ACME Insurance, service-based integration
    - travel insurance, 493–540
  - cataloging, 483
  - CCI (Common Client Interface), J2C, 372
  - centralized orchestration, 290–292
  - centralized policy management, 428–429
  - Certificate Revocation Lists (CRLs), 433
  - change
    - designing XML schemas for, 188
    - service repository for, 484
  - change-time governance, 20, 451, 453–454
  - choreography, of service interactions, 277–278
  - chunking, service payload, 384
  - CICS-based integrations, 557–565
    - ACME’s implementation of, 562–565
    - approaches, 558–562
    - comparison of approaches, 563
    - defined, 556
  - classes, information model, 163–167
  - classes, referenced, 177
  - clusters, information model, 176–177
  - coarse-grained services
    - business services, 57
    - overview of, 53–54
    - service interface design, 205–206
  - COM components, integration with, 557, 565–568
  - Common Client Interface (CCI), J2C, 372
  - Common Object Request Broker Architecture (CORBA), 4–5, 7–10
  - common semantics
    - creating semantic information model, 88–90
    - defining, 40
    - in enterprise policy, 478
    - importance of in SOA, 160–162
    - overview of, 19–20
    - semantic data information type, 52–53
  - communications
    - as architectural principle, 32
    - ESB capabilities, 350
    - specifying service infrastructure, 39
  - compensation
    - implementing for integration services, 382–383
    - service, 296–297
  - Competitive Advantage* (Porter), 125
  - component view, 47, 49
  - components
    - building services with, 102
    - Business Process Model, 139–140
    - reuse using, 12
    - separating security into services and, 416–419
    - services vs., 50–51
  - composite services. *See also* service composition
    - defined, 62
    - evaluating Design Characteristics, 594, 611–612
    - as result of combining services, 274

- composite types
    - analyzing, 597
    - defined, 170
  - computations, business logic, 267
  - conceptual architecture
    - creating project, 226–227
    - travel insurance case study, 497–498, 510–512
  - concerns, separation of, 30
  - conditional flows, 148
  - conditional operation outputs, 148–149
  - conductor-based composition, 280–281
  - confidentiality
    - case study examples, 442, 522
    - defined, 400
    - security and, 400–401
  - configuration
    - defining service, 38
    - evaluating, 596, 617–618
  - connection point, 51
  - consistency, 13–14, 31
  - consistency property, transactions, 382
  - constraints, 97–98, 404
  - construction
    - analyzing, 597
    - business service, 70
    - service, 61
  - context-based routing, 366
  - contrived identifiers, 171
  - conversation state, 218
  - conversational composition services, 279–280
  - CORBA (Common Object Request Broker Architecture), 4–5, 7–10
  - core information modeling. *See* information modeling
  - correlation identifier, 211
  - costs, implementing integration, 356
  - coupling. *See also* loose coupling
    - database-based integration and, 574
    - defined, 64
    - drawbacks of integration using SOI, 360
    - evaluating Design Characteristics, 593, 607–609
    - Russian doll design pattern for, 195–196, 198
    - service interface design, 213–215
  - Course of Action, Business Motivation Model, 135–136
  - covenants, version deployment, 327–328
  - create, read, update, delete (CRUD) services, 386
  - Create Quote operation, 269–272
  - CRLs (Certificate Revocation Lists), 433
  - cross-enterprise access, 397–400
  - CRUD (create, read, update, delete) services, 386
  - cryptography
    - confidentiality and, 400–401
    - digital signatures based on, 404
    - performance impacted by, 446
    - XML Signature relying on, 415
  - customer service, example scenario, 4–7
- D**
- DAC (Discretionary Access Control), 396–397, 400
  - data
    - access control for, 427–428
    - creating semantic information model, 88–90
    - decoupling, 66–67
    - encryption. *See* encryption
    - integration requirements, 15–16
    - islands of, 354–355
  - data access virtualization, Enterprise Data Bus, 386–389
  - data flows, Business Process Models, 139
  - data mapping, 356, 362, 378–380
  - data model reuse, 246
  - data types
    - composite, 170
    - implementing, 170
    - overview of, 167–168
    - simple, 168–169
  - Database Management Systems, J2C adapters for, 372

- databases
  - avoiding SOA stovepipes, 199–200
  - Enterprise Data Bus originating in, 387
  - implementing integration, 375–376, 557, 573–576
  - insurance case study of existing IT, 543–545
  - service management, 342
- data-centric approach processes, 223
- data-passing invocation style, 209–210
- datastores, activity diagram, 257–258
- DCE (Distributed Computing Environment), 8, 66
- DCOM (Distributed Common Object Model), 7–10
- death by architecture, 32–33
- decentralized orchestration, 290–292
- decentralized policy management
  - with attribute propagation, 429–430
  - with identity propagation, 430–431
- decision services, 71–72, 260
- decomposition, 90–94
- deep authentication, 420
- dependencies
  - defining building and use of services, 39
  - loosely vs. tightly coupled, 64
  - managing between service artifacts, 484
  - of security standards, 445
- deploy-time governance
  - defined, 20, 451
  - overview of, 469–471
  - service life cycle, 105
- derivations
  - disallowing, 195–198
  - by extension, 193–194
  - by restriction, 194
  - using abstract classes, 192–193
- derived attributes, 174–175
- design
  - high-level integration, 547–548
  - service discovery during, 22–23
  - SOA best practices, 24–25
  - travel insurance case study, 502–506
- Design Characteristics, evaluating
  - SOA, 605–613
  - abstraction, 606–607
  - composable, 611–612
  - coupling, 607–609
  - governance, 612–613
  - granularity, 609–610
  - isolation of responsibilities, 605–606
  - service evaluation matrix overview, 592–594
  - stateless, 610–611
- Design Perspective, 590–597
- design-time
  - defining building and use of services at, 38
  - developing repository for, 84
  - middle-out process phase of, 112
  - service discovery during, 22–23, 54–55
- design-time governance, 462–469
  - defined, 20, 451
  - overview of, 462–463
  - service design and specification, 465–467
  - service identification, 464–465
  - service implementation, 467–469
  - service policies and business processes, 463
- Desired Results, Business Motivation Model, 134, 135
- determinism, business rules vs. processes, 293
- development
  - determining environment for, 41
  - efficiency in, 14–15, 18
  - model-based, 23–24
- diagrams, defining, 527–528
- digital certificate authentication, 393, 521
- digital signatures, 404, 426
- dimensioned numbers, 168
- dimensions, service, 60–61
- direct database access, integration, 375–376
- direct routing, using service registry, 323–325

- direct trust, 420
  - Directives, Business Motivation Model, 133, 135–136
  - discovery, service
    - during design-time, 22–23
    - dynamic, 54, 55–56
    - specifying service infrastructure, 40
    - using service repository, 483
  - Discretionary Access Control (DAC), 396–397, 400
  - Distributed Common Object Model (DCOM), 7–10
  - Distributed Computing Environment (DCE), 8, 66
  - document model, 248–249, 529–530
  - document-passing invocation style, 208–209, 217
  - documents
    - adapting information model, 179–180
    - Business Process Model, 139, 146–148
    - case study designing, 533–534
    - case study of integration services, 556–557
    - defining, 178–179
    - implementing, 80
    - multiple, 180–181
    - overview of, 177–178
    - service interface design, 221–222, 261–262
  - documents, and XML, 181–190
    - designing for change, 188
    - overview of, 181–183
    - types in schemas, 185–187
    - variations in schemas, 187–188
    - versioning support in schemas, 189–190
    - XML patterns. *See* XML patterns
    - XML schema, 184–185
  - domain data, as information type, 52–53
  - domain object containers. *See* documents
  - domain services
    - characteristics of, 207
    - defining, 62–63
    - granularity of, 57
    - service hierarchy, 58–59, 275–276
    - typical interface combinations, 217
  - domains
    - creating information model from, 163–167, 528
    - defined, 153
    - interoperability levels, 161–162
    - organizing services into, 152–154
    - as point of view, 165
    - service inventory, 156
    - types of, 154–155
  - domain-specific data types
    - composite, 170
    - implementing, 170
    - overview of, 167–168
    - simple, 168–169
  - DSLs (domain-specific languages), 23, 288–289
  - durable property, transactions, 382
  - dynamic binding, 54–55, 56
  - dynamic discovery, 54, 55–56
  - dynamic WS-SecurityPolicy, 436–437
- E**
- EA (Enterprise Architecture)
    - 4+1, services and, 49–50
    - defined, 44
    - reference architecture for SOA, 73–75, 85
    - SOA vs., 44–46
  - EAI (enterprise application integration), 15
  - Eclipse, BPEL Visual Designer for, 301
  - efficiency in development, 14–15, 18
  - element substitution, 196–198
  - encapsulation
    - decoupling interface from implementation, 66
    - with Russian doll design pattern, 196
    - of services, 54
  - encryption
    - confidentiality and, 400–401
    - large messages and, 384
    - XML Encryption standard, 415–416

- endpoint addresses
  - locating services, 98, 321–325
  - security gateway enforcing, 331
  - version deployment using, 328–329
- Ends, Business Motivation Model, 132–133, 134
- enterprise architect, 460
- Enterprise Architecture. *See* EA (Enterprise Architecture)
- enterprise business architecture, 119–120, 123–124
- enterprise business model, 115
- enterprise business processes
  - Business Process Models defining, 64
  - combining services into, 39
  - common patterns of, 69
  - conceptual architecture for, 226
  - defined, 63
  - need for governance, 455–456
  - service granularity for, 56
  - service hierarchy, 58–59
- enterprise business services. *See* enterprise solutions, building
- Enterprise Data Bus, 386–389
- enterprise perspective, of SOA, 37
- Enterprise Resource Planning Systems, 372
- Enterprise Resources and Operational Systems layer, 363
- enterprise resources, as architectural element, 35
- enterprise semantics, 200
- Enterprise Service Bus. *See* ESB (Enterprise Service Bus)
- enterprise service context and inventory, problem model, 232–234
- Enterprise Service Management (ESM), 450–451
- enterprise solutions, building, 311–352
  - applications vs., 312–313
  - architecting security for, 330–333
  - Enterprise Service Bus. *See* ESB (Enterprise Service Bus)
  - example of, 325
  - exception handling and logging, 333–337
  - layered SOA/multitiered application architectures, 317–321
  - locating services, 321–325
  - monitoring and managing, 337–343
  - overview of, 311–312
  - service-based, 313–317
  - versioning, 325–329
- enterprise strategy pattern, 237–238
- Enterprise System Analysis, 107
- enterprise tier, 319, 321
- entitlement, visibility vs., 207
- entity diagram, 525–527
- entity managers, 234
- entity services
  - business logic implementation, 260
  - constructing as service layer, 71–72
  - interaction controller supporting, 314–315
- enumeration data types, 169
- error handling
  - CICS integration, 565
  - enterprise policies for, 478
  - integration with COM components, 568
- ESB (Enterprise Service Bus), 344–351
  - architecture, 346–348
  - choosing, 350–351
  - defining, 344–346
  - engine, 346
  - as framework, 349
  - identity propagation within, 421–422
  - implementing integration access, 377–378
  - as service container, 348–349
  - stand-alone, 348
- ESM (Enterprise Service Management), 450–451
- “Establishing a Service Governance Organization” (Dubray), 450
- EstablishPolicySubmission
  - operation, 545, 548–551, 556
- event-based composition, 285–286
- event-based invocation style, 211–212
- evolution, service repository and, 484
- Excel notation, and data mapping, 379

exception handling  
 building enterprise solutions, 333–337  
 implementing, 266–267  
 service interface design, 220  
 Exception Resolution Service, 336  
 Exceptions/Logging Portal, 336  
 executable Business Process Models, 140–141  
 executable processes  
 BPEL using, 300  
 building with domain-specific languages, 288–289  
 central orchestration with, 292  
 orchestration engine-based  
 composition with, 287  
 turning into abstract processes, 306  
 execution state, 218  
 extended value chain, 125–126  
 extensibility, evaluating, 595–596, 616–617  
 eXtensible Access Control Markup Language. *See* XACML (eXtensible Access Control Markup Language)  
 Extensible Markup Language. *See* XML (Extensible Markup Language)  
 extensions  
 derivation by, 193–194  
 versioning schemas with, 189–190  
 external actors, problem model, 228  
 external services  
 construction of, 61  
 defining, 63  
 granularity of, 57  
 service hierarchy, 58–59, 275–276

**F**

facilitation, as architectural principle, 32  
 failover, 40, 559  
 federated database technology, 387  
 federated identity, 397–400, 413  
 FileNet application, 556, 577–578  
 fine-grained services, 53–54, 205–206  
 fit for purpose, evaluating alignment, 592, 601–602

flexibility  
 accommodating change, 31  
 architectural requirements, 18  
 overview of, 16–18  
 security services for, 418  
 of WS-SecurityPolicy, 436–437  
 foundation domains, 154  
 foundation services  
 business logic, 260  
 conceptual architecture, 227  
 defining, 62  
 granularity, 57–58  
 problem model design, 233  
 service hierarchy, 275–276  
 4+1 Views, 46–50  
 framework, development, 41  
 framework, ESB as, 349  
 functional architect, 459  
 functionality  
 of integration services, 361  
 islands of automation duplicating, 355

**G**

Gang of Four Mediator design pattern, 308  
 gateways, 139, 148–149  
 general-purpose languages (GPLs), 289  
 GeneratePolicyNotice operation, 557  
 GeneratePolicyNoticeDocuments operation, 546, 554–555  
 GetCatastrophicInformation operation, 552–553, 557  
 getDriverRecord operation, 270–271  
 GetDriversInformation operation, 546  
 GetGeoCoding operation, 552–553, 557  
 GetInsurancetoValue operation, 552–553  
 GetInsuranceToValue operation, 557  
 GetPolicyFinancials operation, 557  
 GetPublicProtectionCode operation, 552–553, 557  
 getVehicleInfo operation, 270–271  
 global policy, 428–429, 431–432

- goals
  - Business Motivation Model, 134, 135, 502
  - service interface design, 216–217
- Google Mail, 211
- governance, 449–491
  - defining, 450–452
  - enterprise policy, 477–481
  - evaluating, 594, 612–613
  - metamodel, 74–75
  - middle-out process design, 111
  - need for, 453–459
  - overview of, 20–21
  - reference architecture, 84
  - run-time policies, 486–490
  - of services by policy, 55
  - SOA management and, 450
  - SOA project based on, 115
  - structuring organization for, 475–477
  - types of, 20–21
  - using service repository, 481–486
- GPLs (general-purpose languages), 289
- granularity
  - analyzing, 596
  - business rules vs. processes, 293
  - for business services, 70
  - defined, 53–54
  - defining, 62
  - drawbacks of integration using SOI, 360
  - evaluating, 593–594, 609–610
  - hierarchy of service types and, 56–59
  - integration based on vendor's Web Services, 576
  - integration services and, 361
  - integration using J2C, 373
  - integration with Web Services, 370
  - interface design, 205–207
  - overview of, 60–61
  - specifying size, 38
  - typical interface combinations, 217
- H**
- heterogeneity, and integration, 380
- hierarchical composition services, 279–280
- hierarchical data model, XML, 183, 190
- high-level integration design, 547–548
- history, learning from, 7–10
- Housekeeping Characteristics, evaluating SOA services, 590–591, 619
- HTTP transport, 384, 563
- hubs and spokes, ESB, 345–346
- human activities
  - building enterprise solutions, 313–317
  - manager of, 297–298
  - in service composition, 297–298
- I**
- Identification and Authentication Services, 418
- identifiers, 170–172
- identity
  - authentication and, 394
  - best practices, 419–426
  - federated, 397–400
  - identifiers and, 171
  - travel insurance case study authentication, 521
  - WS-Trust/WS-Federation used for, 408
- Identity Federation Framework (ID-FF), 398, 413
- identity propagation
  - within application server or ESB, 421–422
  - assigning attesting trust for, 422–423
  - case study example, 439, 441–442
  - choosing solution, 425–426
  - decentralized policy management with, 430–431
  - example of, 394–395
  - with REST using Browser-Based SSO, 424–425
  - for Single Sign-On solutions, 420
  - with trusted token service, 423–424
- identity provider, 398
- Identity SSO, 413
- Identity Transformation Services, 418

- ID-FF (Identity Federation Framework), 398, 413
- ID-WSF (Identity Web Services Framework), 398–399, 413
- implementation. *See also* SOA (Service-Oriented Architecture),
  - getting started
  - coupling, 574
  - layers, 535–536
- Implementation Perspective, 590–597
- Influencers, Business Motivation Model, 133, 136
- information
  - islands of automation, 355
  - islands of data, 354
  - islands of security, 355–357
  - role of integration in SOA, 357
- information architecture
  - agility, flexibility, alignment for, 17
  - common semantics for, 19–20
  - creating semantic information model, 88–90
  - designing, 80
  - Enterprise Architecture vs. SOA, 44, 46
  - governance of, 20
  - for services, 52–53
- information hiding, 66, 429
- information metamodel, 73, 75, 84–85
- information modeling
  - association multiplicities, 166–167
  - associations, 166
  - attributes and instances, 165–166
  - business context diagram for, 127–130
  - business model, 223–224
  - classes, attributes and instances, 164–165
  - data types, 167–170
  - derived attributes, 174–175
  - developing from use cases, 201
  - documents, building, 179–180
  - documents, designing, 221–222
  - finding classes, 167
  - identifiers, 170–172
  - objects and attributes, 163–164
  - problem model, 239
  - semantic interoperability using, 163
  - service design process, 506
  - service expectations, 97
  - solution model, 246–248
  - specializations, 172–174
  - structuring, 176–177
  - travel insurance case study, 527–530
  - uniqueness constraints, 170–172
  - value constraints, 176
- initial scenarios, problem model, 229–232
- input parameters, activity diagram, 257–258
- instances, information model, 164–165
- integration, of applications and data, 15–16, 18
- integration access implementation, 365–378
  - data mapping, 378
  - direct database access, 375–376
  - Enterprise Service Bus, 377–378
  - existing Web Services, 369–371
  - JCA/J2C adapters, 372–374
  - Message Broker, 367–369
  - messaging (MOM) infrastructure, 365–367
  - Web Service wrappers, 374–375
- Integration Access layer, 363
- integration components, 260
- integration services, 353–390
  - advantages of, 360–362
  - as architectural layer, 35
  - case study. *See* ACME Insurance, service-based integration
  - challenges of, 354–357
  - characteristics of, 207
  - construction of, 61
  - data mapping for, 378–380
  - data virtualization and Enterprise Data Bus in, 386–389
  - defined, 63
  - Enterprise Service Bus and, 350
  - granularity of, 57
  - interface combinations, 217
  - large messages and, 384–386

- layered enterprise architecture for, 363–364
- on-demand, 100–102
- overview of, 358–360
- reference architecture for, 84
- security for, 380–381
- service hierarchy and, 58–59, 275–276
- transactional support, 381–383
- versioning, 383–384
- integrity
  - defined, 400
  - security and, 401–403
  - travel insurance case study, 522–523
- interaction controller, 313–315
- interaction model
  - business context diagram, 127, 508
  - service expectations, 97
- interaction style, 207–213
  - analyzing, 597
  - data passing, 209–210
  - document passing, 208–209
  - events, 211–212
  - mixed styles, 212–213
  - overview of, 207–208
  - parameter passing, 208
  - request/reply, 210–211
- interactions, service, 530
- interceptor, 331–333
- interceptor agent, 343
- intermediary-based routing, 324–325, 328
- Internet banking example, 5–6
- Internet scope, service interface, 206
- interoperability, 67–68
- invocation
  - implementing privacy, 355
  - of services by consumers, 321–325
  - state, 218
- islands of automation, 355
- islands of data, 354–355
- islands of security, 355–357
- isolated property, transactions, 382
- isolation of responsibilities
  - evaluating, 592, 605–606
  - services and, 54
  - Two-Phase Commit protocol, 295–296
- ivory tower dictator, 477
- J**
  - J2C (J2EE Connector Architecture), 372–374, 381
  - J2EE applications, integration with, 557, 570–573
  - Java
    - integration with existing APIs, 557, 568–570
    - limitations for enterprise applications, 9
  - Java-COM bridge, 566–567
  - JCA (Java Connector Architecture)
    - adapters, 372–374, 558
  - JMS (Java Message Service), 570–571
  - JNI (Java Native Interface), 566–567
  - JSR (Java Specification Request), 316
- K**
  - Kerberos
    - Enterprise Service Bus support for, 350
    - governance policy, 452
    - identity propagation to COM
      - components, 568
    - integration implementation, 380, 565
    - security architecture supporting, 330
    - WS-Security Token Profile, 406, 577
  - knowledge, loose coupling
    - requirements, 68
  - KPIs (key performance indicators), 78, 341–342
- L**
  - lanes, Business Process Models, 140
  - languages
    - domain-specific, 23, 288–289
    - general-purpose, 289
    - orchestration, 276
    - Rights Expression Language, 406
    - SAML. *See* SAML (Security Assertion Markup Language)

- languages (*continued*)
    - Web Ontology Language, 162
    - Web service, 277
    - Web Services Choreography
      - Language, 277–278, 286–290
    - WS-BPEL. *See* WS-BPEL (Web Services Business Process Execution Language)
    - WSDL. *See* WSDL (Web Services Description Language)
    - XACML. *See* XACML (eXtensible Access Control Markup Language)
    - XML. *See* XML (Extensible Markup Language)
  - large size messages, 384–385
  - layered SOA architecture, 316–321
  - learning from history, 7–10
  - legacy systems
    - building services from, 99
    - functionality of integration services, 361
    - integrating into service environment, 39
    - integration using J2C, 373
    - integration using Web Services wrappers, 375
    - integration with COM components, 565–568
  - legal values, 165
  - legalities, SOA governance, 453
  - LEGO analogy, 42–43
  - Liberty Alliance, 398, 413
  - life cycle, service, 104–106
  - line-of-business services. *See* LOB (line-of-business) services
  - load balancing, CICS integration, 559, 563
  - LOB (line-of-business) services
    - conceptual architecture for, 226
    - granularity, 57
    - service hierarchy, 58–59
  - local policies, combining global and, 431–432
  - locating services, 98, 321–325
  - location coupling, 574
  - location transparency
    - loose coupling for, 65–66
    - services designed for, 55
    - specifying service infrastructure, 40
  - logging, exception
    - CICS integration using, 565
    - integration with COM components, 568
    - overview of, 336–337
  - Logging Service, 336
  - logic implementation, 259–260
  - logical view, 47, 49
  - login, username/password, 393
  - loose coupling, 64–68
    - assumptions and, 68
    - data and, 66–67
    - between interface and implementation, 66
    - between interoperability and platform independence, 67–68
    - knowledge and, 68
    - location transparency and, 65–66
    - overview of, 64–65
    - separating security into
      - services/components, 416–419
    - service interface design using, 213–215
    - between services, 54, 274
    - Two-Phase Commit protocol and, 296
    - usage and, 68
    - versioning and, 67
- M**
- MAC (Mandatory Access Control), 396–397, 400
  - major changes, XML schemas, 188
  - management, 337–343
    - business activity, 338–340
    - Enterprise Service Bus, 350–351
    - overview of, 337–338
    - SOA governance vs. SOA, 450–451
    - SOA tools for, 473
    - technical, 340–343
  - Mandatory Access Control (MAC), 396–397, 400

- mashups, 274–275
  - maturity levels, semantic
    - interoperability, 161
  - MBD (model-based development), 74–75, 85
  - Means, Business Motivation Model, 132–136
  - mediator
    - conductor-based composition using, 280–281
    - data transformation using, 378–379
    - ESB support for, 347
    - routing/transformation problems using, 328
  - medium-grained services, 57
  - memory utilization, 384
  - MEPs (Message Exchange Patterns), 208
  - Message Brokers, 367–369
  - Message Exchange Patterns (MEPs), 208
  - message sender
    - best practices, 419
    - defined, 394
    - integrity and, 402
    - non-repudiation, 402
    - WS-Trust, 407–408
  - messages
    - business context diagram, 126
    - ESB processing capabilities, 350
    - implementing data mapping as part of delivery, 378
    - large size, 384–386
  - Message Oriented Middleware (MOM)
    - infrastructure, 365–368
  - messaging systems, J2C adapters for, 372
  - metadata, 527–528
  - metrics
    - creating reference architecture, 84–85
    - measuring success with, 41
  - middle-out process, service design, 109–112
  - middleware abstraction layers, CICS
    - integrations, 558–559
  - minimum architecture, 82–84
  - minor changes, XML schemas, 188
  - Mission, Business Motivation Model, 133, 502
  - model-based development (MBD), 74–75, 85
  - modeling. *See also* information modeling
    - business context diagram, 506–508
    - Business Process Models. *See* Business Process Models
    - business processing, 22
    - defining, 527–528
    - development based on, 23–24
    - ESB capabilities, 351
    - first rules of, 508
    - interaction model, 508
    - shared information model, 508
  - modernization, of existing applications, 99
  - modularity of services, 53–54
  - MOM (Message Oriented Middleware)
    - infrastructure, 365–368
  - monitoring, of enterprise solutions
    - business activity, 338–340
    - ESB support for, 347
    - overview of, 337–338
    - technical monitoring of SOA solutions, 340–343
  - monitoring agents, types of, 342–343
  - motivation, business. *See* BMM (Business Motivation Model)
  - MQ infrastructure
    - CICS integration, 559–561, 563
    - dealing with large messages, 384
  - multiple population identifiers, 171–172
  - multitiered application architecture, 317–321
  - MVC (Model-View-Controller) design pattern, 313–314, 416–419
- N**
- namespaces, XML, 189–190, 197–198
  - naming conventions, 478, 530
  - NetBeans, 301
  - .NET/COM bridge, 565–568

nonpersistent messaging, 366  
 non-repudiation  
   case study examples, 442, 522–523  
   defined, 400  
   security and, 404–405  
 Notification Service, 336  
*n*-tier solution architecture, 318–321,  
   518–519  
 numeric data types, 168

**O**

Object Management Group (OMG), 132  
 Objectives, Business Motivation Model  
   applying via Directives, 135–136  
   case study, 502  
   overview of, 134  
 objects  
   achieving reuse through, 12  
   derived attributes from, 174–175  
   identifiers and uniqueness constraints  
     of, 170–172  
   information model defining,  
     163–167  
   modeling using specializations,  
     172–174  
   services vs., 50  
 OMG (Object Management Group),  
   132  
 on-demand services, 100–102  
 OO (object-oriented) design  
   overview of, 90–91  
   use of, 164  
   XML hierarchical data model vs., 183,  
     190  
 OpenCSA (Open Composite Services  
   Architecture), 283  
 operation procedures  
   case study, 536–538  
   implementation of services, 224  
   service definition diagrams, 244–245  
   within solution model, 246  
 operational logic, modeling, 257–258  
 operational systems, as architectural  
   element, 35  
 (optional) attribute, XML schemas, 189

orchestration  
   with BPEL, 299–301  
   centralized and decentralized,  
     290–292  
   defined, 36, 63  
   engine-based composition, 286–290  
   human activities incorporated into,  
     297–298  
   overview of, 276–278  
   pitfalls of service composition,  
     307–308  
   supporting business rules, 293–294  
 output parameters, activity diagram,  
   257–258  
 outsourced (rented) services,  
   99–102  
 OWL (Web Ontology Language), 162  
 ownership  
   analyzing, 596  
   business service, 70  
   service, 60–61  
   structuring for SOA governance, 476

**P**

packaged systems, integrating, 39  
 PADBAC (Predetermined  
   Authorization Decision-Based  
   Access Control), 397, 400,  
   432–434  
 PAPs (Policy Administration Points)  
   authorization for access control using,  
     395  
   policy application points vs., 486  
   XACML architecture, 414  
 parameter-passing invocation style,  
   208, 217  
 partitioning, domain, 152  
 passwords  
   authentication, 393  
   case study, 437–439  
   WS-Security Username Token Profile,  
     406  
 pattern, as architectural principle, 31  
*Patterns of Enterprise Application  
 Architecture* (Fowler), 301

- PDPs (Policy Decision Points)
  - authorization for access control using, 395–396
  - centralized policy management using, 428–429
  - combining global and local policies, 431–432
  - decentralized policy management using, 429–431
  - Mandatory Access Control using, 396
  - WS-Federation using, 409
  - XACML architecture using, 395
- peer-to-peer based composition, 280–281
- PEPs (Policy Enforcement Points)
  - authorization for access control using, 395–396
  - centralized policy management and, 429
  - decentralized policy management with attribute propagation, 429–430
  - decentralized policy management with identity propagation, 430–431
  - enforcing service policies at run-time, 471
  - integration based on vendor's Web Services, 578
  - Mandatory Access Control using, 396
  - run-time policy enforcement using, 486–488
  - WS-Federation using, 409
  - XACML architecture, 395
- performance
  - building enterprise solutions, 314
  - centralized policy management and, 429
  - combining global and local policies, 431–432
  - database-based integrations and, 574–575
  - impact of security on, 446
  - persistent messaging, MOM implementations, 366
  - phases, implementing security in, 444
  - physical data, as information type, 52–53
  - physical view, 48
  - PIPs (Policy Information Points), 395
  - planning, for security, 443–444
  - platform independence, 67–68
  - platform profiles, reference architecture, 74–75
  - point-and-click WSDL generation, 199–200
  - point-to-point authentication, 425–426
  - policies
    - Business Motivation Model, 502
    - defined, 55
    - design-time governance. *See* design-time governance
    - developing and registering run-time, 486–488
    - developing enterprise, 477–481
    - directives governing business, 135–136
    - enforcing and adapting run-time, 488–490
    - governance of, 20, 105–106, 450–454, 477–481
    - integration services case study, 548–551, 555–556
    - middle-out process design, 111
    - monitoring enterprise solutions, 338
    - need for explicit run-time service policies, 456–457
    - run-time enforcement and adaptation, 488–490
    - separating business logic from, 457–458
    - separating business processes from service, 463
    - service constraints, 97–98
    - versioning, 66
  - Policy Administration Points. *See* PAPs (Policy Administration Points)
  - policy application points, 486–488
  - Policy Decision Points. *See* PDPs (Policy Decision Points)

- policy enforcement approaches, 428–435
    - choosing solution, 434–435
    - combining local and global enterprise policy, 431–432
    - decentralized PDP/PEP, 430–431
    - predetermined decision-based models, 432–434
    - purely centralized PDP, 428–429
    - purely decentralized PDP/PEP, 429–430
  - Policy Information Points (PIPs), 395
  - Policy Management Service, case study, 440
  - policy push method, 431–432
  - policy retrieval method, 431
  - Policy Retrieval Service, 418, 440
  - portals, 315–316
  - portfolio architect, 460
  - portfolio rationalization, 78–79
  - Potential impact, 136
  - powerless committee, 477
  - Predetermined Authorization
    - Decision-Based Access Control (PADBAC), 397, 400, 432–434
  - PriceForQuote operation, 269–270
  - PriceRequest document, 270–271
  - primary activities, extended value chain, 125–126
  - principal domains, 154
  - principles, architectural, 30–33
  - privacy, pseudonym SSO for, 413
  - private processes, 316
  - problem space model, 227–241
    - actors, 228–229
    - describing, 224
    - detailed scenario diagrams, 234–238
    - enterprise service context and inventory, 232–234
    - information model, 239
    - initial scenario diagrams, 229–232
    - service specifications, 239–241
    - use case diagrams, 227–228
  - process (behavior) model, 97
  - process documentation structure, 225
  - process flows, Business Process Models, 139
  - process metamodel, reference architecture, 74–75
  - process steps, Business Process Models, 139
  - process view, 48–50
  - process-centric approach processes, 223
  - processes
    - business. *See* business processes
    - Business Process Models and, 149–150
    - modeling in service composition, 303–307
    - organizing service inventory for enterprise, 232
  - programmatically composition, 281–282
  - programming, DSLs vs. GPLs, 289
  - project business architecture
    - defined, 120
    - enterprise business architecture vs., 123–124
    - features of, 124–125
  - project-specific interoperability, 161
  - proxy agent, 343
  - pseudonym SSO, 413
  - public key cryptography, 404, 415
  - publish-find-bind triangle, 456–457, 486–487
  - Publish/Subscribe (Pub/Sub)
    - composite service, 285–286
  - pull model, 298
  - push model, 298
- Q**
- QoS (quality of service)
    - ESB and, 350
    - service specifications and, 94
  - QuoteRequest document, 269
- R**
- Radio Frequency Identification (RFID)
    - reader, 211
  - RAS (Reusable Asset Specification), 485–486
  - RateAutoPolicy integration, 551
  - RateCommercialAutoProperty
    - operation, 557
  - RateCommercialProperty operation, 551, 556

- RateInsurancePolicy operation, 546, 551–552
- The Rational Unified Process — An Introduction* (Booch and Krutchen), 28
- RBAC (Role-Based Access Control), 397, 400
- RDF (Resource Description Framework), 162
- recursive aggregation, of orchestration engines, 288
- reference architecture
  - contents of, 73–75
  - getting started, 82–85
  - goals of, 73
  - methodology, 80–81, 82–85
  - middle-out process phase, 111
  - overview of, 19
- referenced classes, 177
- references, document marking, 248
- registry. *See* service registry
- ReinstateInsurancePolicy operation, 545, 551, 556
- REL (Rights Expression Language), 406
- relying party, 398
- Remote Method Invocation (RMI), 566–567, 570
- Remote Procedure Call (RPC), 8
- reporting, reference architecture, 85
- repositories, middle-out process design, 111
- Representational State Transfer. *See* REST (Representational State Transfer)-based Web services
- request/reply invocation style, 210–211
- request/response protocol
  - SAML, 412
  - XACML, 413–414
- requirements, security, 443–444
- research, integration and, 356
- Reservation Web Service, 442
- resource access layer
  - business logic implementation, 260
  - implementation components, 259–260
  - implementing, 267–268
  - overview of, 254–256
  - responsibilities, 256–257
- Resource Description Framework (RDF), 162
- resource tier, 319–320, 321
- REST (Representational State Transfer)-based Web services
  - confidentiality, 401
  - identity propagation, 424–425
  - integrity, 403
  - non-repudiation, 404
- restriction, derivations by, 194
- retirement, service, 105, 473–475
- RetrieveInsurancePolicy operation, 546, 551, 556
- RetrievePolicyComplianceInformation operation, 546, 557
- return on investment (ROI), of SOA, 273
- Reusable Asset Specification (RAS), 485–486
- reuse
  - analyzing, 597
  - architectural requirements, 13–14, 18
  - challenges of, 11–13
  - data model, 246
  - DSLs vs. GPLs, 289
  - example scenario, 6
  - integration policy for, 362
  - motivations for, 10–11
  - need for governance, 455
  - promoting consistency, 31
- Reusable Asset Specification, 485–486
  - of services, 54
- revisions, XML schemas, 188
- RFID (Radio Frequency Identification) reader, 211
- Rights Expression Language (REL), 406
- RMI (Remote Method Invocation), 566–567, 570
- roadmap, reference architecture, 83–85
- ROI (return on investment), of SOA, 273
- Role-Based Access Control (RBAC), 397, 400
- root class, selecting for document, 178
- root node, document marking, 248

RPC (Remote Procedure Call), 8  
 rules. *See* business rules  
 rules engines, 294–295  
 run-time  
   design-time registry vs., 22  
   ESB configuration at, 347  
   governance, 451, 471–475  
   tracking what is running at, 38  
 run-time service policy  
   analyzing, 597  
   authoring, 478  
   creating, 451  
   design phase, 462  
   developing and registering, 486–488  
   enforcement and adaptation, 488–490  
   need for explicit, 456–457  
   run-time phase, 471–475  
 Russian doll, XML design pattern,  
 195–196

**S**

SaaS (software-as-a-service), 85,  
 100–102  
 Salami Slice, XML design, 196–197  
 SAML (Security Assertion Markup  
 Language)  
   Browser-Based SSO for REST,  
   424–425  
   case studies using, 441–442, 521  
   defined, 408  
   overview of, 411–413  
   Token Profile standard, WS-Security,  
   406, 420, 441–442  
   using federated identity, 398–399  
   WS-Federation using, 409  
   WS-Trust accommodating, 407–408  
   XACML policies carried by, 414–415  
 SAML Issuing Authority, 412  
 SCA (Service Component Architecture)  
   building services using, 102  
   composition, 282–285  
   implementing business components,  
   362  
 scalability, centralized orchestration  
   and, 290–292  
 scenarios, use case  
 Business Process Models, 144  
 case studies, 496–497, 514–517  
 creating information model, 528  
 implementing business layer,  
   268–272  
 multiple, 144–146  
 problem model diagrams, 229–232,  
   234–238  
 service design, 505–506  
 service interface design, 223–224  
 service model, 241–243  
 step reuse, 146  
 scope  
   analyzing, 596  
   business service, 69  
   service, 60–61  
   service interface design, 205–207  
   software architecture vs. SOA, 31  
   typical interface combinations, 217  
 security, 391–447  
   access control, 395–397, 427–435  
   auditing, 435–436  
   authentication, 392–395, 419–426  
   authorization, 395–396  
   case study examples, 437–442,  
   523–524  
   CICS integration, 564–565  
   complete architecture analysis of, 437  
   components. *See* components  
   confidentiality, 400–401  
   credentials, 380  
   cross-enterprise access, 397–400  
   database-based integrations,  
   575–576  
   ESB support for, 347, 350  
   evaluating Alignment Characteristics,  
   592, 602–603  
   federated identity, 397–400  
   flexibility with WS-SecurityPolicy,  
   436–437  
   gateway, 330–331  
   high-level game plan for, 443–447  
   identity, 419–426  
   integration role, 357  
   integration support, 380–381  
   integration using J2C, 372

- integration using vendor's Web Services, 577–578
- integration with COM components, 566, 568
- integration with MOM, 366
- integrity, 401–403
- interceptors, 331–333
- islands of, 355–357
- non-repudiation, 404–405
- overview of, 330
- selecting enterprise service products, 416–419
- tagging, 427–428
- terminology, 400–401
- troubleshooting, 435–436
- Web Services. *See* Web Services, standards and specifications
- Security Assertion Markup Language. *See* SAML (Security Assertion Markup Language)
- Security Token Service. *See* STS (Security Token Service)
- SEI (Software Engineering Institute), 28
- self-describing, service contracts, 55
- Semantic Alignment, evaluating, 592, 603–605
- semantic data information type, 52–53
- semantic information model
  - creating, 88–90
  - implementation of, 78
  - reference architecture, 83
  - SOA methodology, 81
- semantic input validations, 263–264
- semantic interoperability
  - avoiding SOA stovepipes, 199–200
  - core information modeling, 163–167
  - importance of, 160–162
- The Semantic Web: A Guide to the Future of XML, Web Services, and Knowledge Management* (Daconta, Obrst, and Smith), 161, 163
- "The Semantic Web", *Scientific America*, 162–163
- semantics, common. *See also* service context and common semantics
  - creating semantic information model, 88–90
  - defining, 40
  - in enterprise policy, 478
  - importance of, 160–162
  - overview of, 19–20
  - semantic data information type, 52–53
- SEMCI (Single Entry Multiple Carrier Interface), 225
- sender-voucher confirmation method, 421
- separation of concern
  - applying to service types, 70–72
  - as architectural principle, 30
  - logical design vs. technology implementation, 49
- servers, centralized, 290–292
- service architecture, basic, 254–260
  - activity diagrams, 257–258
  - implementation components, 259–260
  - layer responsibilities, 256–257
  - overview of, 254–256
- service autonomy. *See* autonomy, service
- service business layer
  - implementation components, 259–260
  - implementing, 263–267
  - implementing, example of, 268–272
  - implementing interface layer, 260
  - overview of, 254–256
  - responsibilities, 256–257
- Service Component Architecture. *See* SCA (Service Component Architecture)
- service composition, 273–309. *See also* composite services
  - architectural models in, 279–281
  - avoiding static, programmatic orchestration, 307–308
  - BPM-composition relationship, 278–279

- service composition, 273–309. *See also*
  - composite services (*continued*)
  - business rules and, 292–295
  - case study example, 301–307
  - incorporating human activities into, 297–298
  - orchestration and choreography, 276–278
  - orchestration with BPEL, 299–301, 308–309
  - separation into service layers, 275–276
  - transactions and, 295–297
  - understanding, 274–275
  - using layered service approach, 308
- service composition, implementing, 281–292
  - centralized/decentralized approaches, 290–292
  - event-based approach, 285–286
  - orchestration engine-based approach, 286–290
  - programmatic approach, 281–282
  - Service Component Architecture approach, 282–285
- service container, ESB as, 348–349
- service context and common semantics, 159–202
  - best practices, and pitfalls, 199–201
  - core information modeling, 163–167
  - defining types, 167–170
  - derived attributes, 174–176
  - documents, 177–181
  - documents and XML. *See* documents, and XML
  - identifiers and uniqueness constraints, 170–172
  - importance of semantics, 160–163
  - specializations, 172–174
  - structuring information models, 176–177
  - value constraints, 176
  - XML patterns. *See* XML patterns
- service contracts, 55, 260
- service definitions, 243–246, 350
- service deployment, governance, 469–471
- service design, 106–109
  - bottom-up approaches, 108–109
  - design-time governance, 465–467
  - overview of, 106
  - service life cycle, 105
  - top-down approaches, 106–108
- service endpoint addresses
  - interceptors enforcing, 332
  - locating services, 98, 321–325
  - security gateway enforcing, 331
  - version deployment using, 328–329
- service execution model, 97
- service hierarchy, 56–59, 275–276, 308
- service identification
  - design-time governance, 464–465
  - overview of, 90–94
  - service life cycle, 105
  - summary of, 102–103
- service implementation design, 253–271. *See also* business use cases
  - basic service architecture, 254–260
  - business layer, 263–267
  - business layer, example, 268–272
  - case study, 534–538
  - governance, 467–469
  - interface layer, 260–262
  - overview of, 253–254
  - resource layer, 267–268
- service infrastructure specialist, 461
- service interface
  - building, 38
  - creating reference architecture, 83
  - decoupling from implementation, 66
  - defined, 62
  - future of semantic, 89–90
  - governance of, 22
  - implementation components, 259–260
  - implementing, 260–262
  - integration of applications/data with, 15–16
  - overview of, 50–52, 254–256
  - responsibilities, 256–257
  - service level agreement and, 53

- service interface design, 203–252
  - business use cases. *See* business use cases
  - case study, 530–532
  - document design, 221–222
  - example of. *See* ACME Insurance, service interface design
  - exceptions, 220
  - identifying granularity, 217
  - interaction styles, 207–213
  - isolating responsibilities, 213–215
  - service characteristics and, 204–207
  - SOA context and, 204
  - stateless interfaces, 218–220
  - summary of, 249–251
  - understanding overall context, 215–216
- service inventory
  - designing problem model, 232–234
  - designing SOA, 155–156
  - service identification, 93–94
  - service interface design, 216–217
  - travel insurance case study, 524–525
- service level agreements. *See* SLAs (service level agreements)
- service life cycle, SOA governance, 459–475
  - deploy-time, 469–471
  - design-time, 462–469
  - overview of, 104–106, 459–461
  - phases of, 451–452
  - run-time, 471–475
- service management
  - architecture, 341–342
  - exception logging, 337
- service metamodel, reference architecture, 73, 75
- service model, 6–7, 241–243
- Service Provider Interface (SPI), J2C, 372
- service registry
  - CICS integration using, 564
  - ESB support for, 346
  - implementing location transparency, 65–66
  - overview of, 323–324
- publishing run-time policies in, 487–488
- role of, 322
- service repository
  - artifacts, 484–485
  - basic architecture, 481–483
  - cataloging and discovery, 483
  - dependency management, 484
  - governance using, 481–486
  - Reusable Asset Specification, 485–486
  - service evolution and versioning, 484
  - validation, 483
- service retirement, 105, 473–475
- service specifications
  - constraints, 97–98
  - current practices, 95–96
  - evaluating, 595
  - expectations, 96–97
  - formal vs. informal, 241
  - interaction model of, 97
  - location of, 98
  - overview of, 94
  - problem model of, 239–240
  - service life cycle and, 105
  - travel insurance case study, 534–535
- service tester, 461
- service utilization process, run-time governance, 471–474
- service-based enterprise solutions, 313–317
- service-based integration case study. *See* ACME Insurance, service-based integration
- Service-Oriented Architecture
  - getting started. *See* SOA (Service-Oriented Architecture), getting started
  - promise of. *See* SOA (Service-Oriented Architecture), promise of
- Service-Oriented Enterprise (SOE), 450
- Service-Oriented Integration (SOI), 358–360
- services
  - analysis and design, 24–25
  - business. *See* business services

- services (*continued*)
  - Business Process Models and, 149–151
  - buying, 98–99
  - conceptual architecture, 497–498
  - design process, 505–506
  - design-time discovery, 22–23
  - efficiency in developing, 15
  - enterprise solutions. *See* enterprise solutions, building
  - governance of, 20, 22
  - granularity. *See* granularity
  - implementation of, 79–81
  - integration. *See* integration services
  - organizing, 151–155
  - reference architecture, 83–85
  - reuse of. *See* reuse
  - security, 416–419
- services, characteristics of
  - common patterns, 68–70
  - dimensions, 60–64
  - granularity, 56–59
  - loose coupling, 64–68
  - overview of, 53–56
  - types and purpose, 70–72
- services, evaluating SOA, 589–619
  - Alignment Characteristics, 597–605
  - Design Characteristics, 605–613
  - evaluation matrix, 590–597
  - Housekeeping Characteristics, 619
  - overview of, 589–590
  - Technical Characteristics, 613–619
- services, SOA architecture
  - fundamentals, 37–72
  - aligning to business, 40–41
  - aligning with business, 40–41
  - building and using, 38–39
  - characteristics of. *See* services, characteristics of
  - combining into enterprise processes, 39
  - defining, 37–38, 50–52
  - defining common semantics and data, 40
  - definitions, 61–64
  - enterprise architecture for, 44–46
  - information architecture relating to, 52–53
  - integrating packaged and legacy systems, 39
  - specifying technology infrastructure, 39–40
- services assembler, 461
- services librarian, 461
- services metamodel, 84
- services realization, 98–104
  - building, 102–103
  - buying, 99
  - outsourcing, 99–102
  - overview of, 98–99
  - summary of, 103–104
- Session Façade J2EE design pattern, 308
- shared information model
  - business context providing, 127–130, 508
  - deriving documents from, 221–222
- silent rollback, of 2PC, 296
- simple data types, 168–169
- Single Entry Multiple Carrier Interface (SEMCI), 225
- Single Sign-On. *See* SSO (Single Sign-On)
- size, large message, 384–386
- SLAs (service level agreements)
  - analyzing, 597
  - defined, 53
  - designing, 343
  - design-time governance, 462
  - ESB capabilities, 350
  - evaluating Technical Characteristics, 595, 615–616
  - service managers evaluating, 341–342
- smart data continuum, 161
- SOA (Service-Oriented Architecture), getting started
  - business architecture, 85–86
  - business processes, 86–88
  - compromise approach, 109–112
  - identifying services, 90–94, 102–103
  - information design, 88–90
  - methodology overview, 78–82

- practical steps, 113–115
- realizing services, 98–103
- reference architecture, 82–85
- service design process, 106–109
- service life cycle, 104–106
- specifying services, 94–98
- SOA (Service-Oriented Architecture),
  - promise of
  - agility and flexibility, 16–18
  - alignment, 17–18
  - analysis and design, 24–25
  - business processing modeling, 22
  - common semantics, 19–20
  - design-time service discovery, 22–23
  - efficient development, 14–15, 18
  - example scenario, 4–7
  - governance, 20–22
  - integration of applications and data, 15, 18
  - learning from history, 7–10
  - model-based development, 23–24
  - motivations for using, 10–11
  - reference architecture, 19
  - reuse, 11–14, 18
- SOA run-time architect, 461
- SOA stovepipes, 199
- “The social side of services”, *IEEE Internet Computing* (Vinoski), 322
- SOE (Service-Oriented Enterprise), 450
- software abstraction layers, CICS
  - integrations, 558–559
- software architecture
  - comparing SOA to, 46–48
  - defining, 28–29
  - principles and practices of, 30–33
  - SOA vs., 31
  - styles, 29–30
- Software Engineering Institute (SEI), 28
- software-as-a-service (SaaS), 85, 100–102
- SOI (Service-Oriented Integration), 358–360
- solution lead, 459
- solution model, service interface
  - design, 241–249
  - describing, 224
  - document model, 248–249
  - information model, 246–248
  - operations procedures, 246
  - overview of, 222–223
  - service definition diagrams, 243–246
  - service model, 241–243
- specializations, 172–174
- specifications
  - analyzing, 597
  - evaluating Alignment Characteristics, 591, 599–601
  - evaluating Technical Characteristics, 613–615
- SPI (Service Provider Interface), J2C, 372
- SSL authentication
  - confidentiality and, 522
  - integrity and, 522
  - point-to-point authentication using, 425–426
  - travel insurance case study, 521
- SSL/TLS authentication protocols, 393, 400–403
- SSO (Single Sign-On)
  - browser-based vs. service-based, 398
  - case study, 438–439
  - defined, 355
  - SAML 2.0 support for, 413
  - using federated identity for, 397–398
- SSO (Single Sign-On), identity
  - propagation for, 420–426
  - within application server or ESB, 421–422
  - assigning attesting trust, 422–423
- Browser-Based SSO for REST, 424–425
- choosing solution, 425–426
- defined, 394–395
- overview of, 420–421
- using trusted token service, 423–424
- staff resolution, 297–298
- stakeholders
  - role in SOA governance, 459–460
  - service deployment, 469–471
  - service design and specification, 465–467

- stakeholders (*continued*)
    - service identification, 464–465
    - service implementation, 467–469
    - service retirement, 475
    - service utilization, 473–474
  - stand-alone ESB architecture, 348
  - standards
    - developing enterprise policy, 478
    - fully understanding details of, 445
    - reusing, 200–201
    - selecting products for enterprises
      - based on, 419
      - using accepted, 444–445
  - statefulness, 293
  - stateless
    - evaluating, 594, 610–611
    - service operations, 55
  - stateless service interfaces, 218–220
  - step reuse, 146
  - store-and-forward pattern, 385–386
  - stored procedures, 376
  - Strategy, Business Motivation Model
    - applying via Directives, 135–136
    - defined, 134
    - implementing Goals through, 135
    - travel insurance case study, 502
  - structuring organization, for
    - governance, 475–477
  - STS (Security Token Service)
    - as authorization service for
      - WS-Federation, 409
    - case study, 440–442
    - defined, 418
    - defined by WS-Trust, 406–408
    - WS-SecureConversation, 410
  - styles, architectural, 29–30
  - subelements, document marking, 248
  - subjects, business context diagram, 126
  - subpopulation identifiers, 172
  - supporting activities, extended value
    - chain, 125–126
  - symbolic data types, 169
  - synchronicity, business rules vs.
    - processes, 293
  - synchronous invocations, 373
  - syntactic coupling, 160
  - syntactic data validations, 260–263
  - system exceptions, service interface
    - design, 220
- T**
- Tactics, Business Motivation Model, 135, 502
  - task services
    - business logic implementation, 260
    - constructing as service layer, 71–72
    - interaction controller supporting, 315
  - Technical Characteristics, 613–619
    - autonomy, 618–619
    - extensibility, 616–617
    - service evaluation matrix overview, 595–596
    - service level agreement, 615–616
    - services, evaluating SOA, 613–619
    - specification, 613–615
    - variability and configurability, 617–618
  - technology
    - agility, flexibility and alignment of, 17
    - analysis and design of, 24
    - business architecture, 122–123, 130–132
    - EA and SOA architecture, 45–46
    - implementing integration and, 356
    - logical design vs. implementation of, 49
    - monitoring SOA solutions, 340–343
    - reference architecture for SOA, 74–75
    - service specifications, 96
    - specifying service infrastructure, 39–40
  - testing, middle-out process phase, 112
  - TFIM (Tivoli Federated Identity Manager), 381, 565
  - 3-tiered application architecture
    - n*-tiered vs., 318–321
    - overview of, 317–318
  - Tivoli Federated Identity Manager (TFIM), 381, 565
  - Token Profile standards, WS-Security, 406, 412, 420
  - tokens, SAML, 412

- tokens, trust propagation using, 420
  - top-down SOA, service design, 106–107, 109–113
  - traceability, Business Motivation Model, 136–137
  - Transaction Monitors, J2C adapters for, 372
  - transactions
    - ESB support for, 346
    - integration using J2C, 372
    - MOM-based integration, 366
    - service composition and, 295–297
    - support in integration, 381–383
  - transformations
    - components of, 259
    - implementing data, 379–380
    - implementing interface layer, 262
    - reference architecture for SOA, 74–75
  - transitive trust, 420–423
  - transport protocols, 380
  - travel insurance case study, 493–540
    - analysis and design review, 502–506
    - authentication, 519–521
    - authorization, 521–522
    - business analysis, 506–508
    - business concerns, 498–502
    - business process model, 509–510
    - conceptual architecture, 497–498
    - confidentiality, 522
    - document design, 533–534
    - entity diagram, 525–527
    - information model, 527–530
    - integrity and non-repudiation, 522–523
    - scenario, 496–497
    - security design, 523–524
    - service conceptual architecture, 510–512
    - service implementation design, 534–538
    - service interface design, 530–532
    - service inventory, 524–525
    - solution architecture, 518–519
    - use cases, 512–517
  - troubleshooting, auditing and, 435–436
  - trust, 425–426. *See also* identity propagation
  - trusted token service, 423–424
  - try/catch blocks, exception handling, 266, 333–337
  - Tuxedo, 7–10
  - 2PC (Two-Phase Commit) protocol, 295–296
  - type, analyzing, 597
- U**
- UDDI (Universal Description, Discovery and Integration) registry, 323
  - undo actions, of 2PC, 296
  - uniqueness constraints, 170–172
  - Universal Description, Discovery and Integration (UDDI) registry, 323
  - UpdateInsurancePolicy operation, 545, 550, 556
  - URI mapper, 561–562
  - usage
    - analyzing, 597
    - loose coupling for, 68
    - loose coupling requirements, 68
    - service life cycle, 105
    - service pattern, 71
  - use cases. *See also* business use cases
    - Business Process Models and, 144–146
    - creating information model from, 528
    - designing initial scenarios, 229–232
    - developing information models based on, 201
    - identifying for service interface design, 223–224, 225
    - overview of, 48, 143–144
    - problem model, 227–228
    - service design process, 506
    - step reuse, 146
    - travel insurance case study, 512–517
  - <<used>> and <<used by>> relationship, coupling, 64
  - user identity, 380–381, 413
  - user tier, 318–319, 320
  - user-facing mashups, 275

- usernames
  - authentication, 393
  - case study, 437–439
  - WS-Security Username Token Profile, 406
- utility services
  - as bottom-up approach, 108
  - characteristics of, 207
  - defined, 63
  - granularity of, 57–59
  - service hierarchy and, 275–276
  - typical interface combinations, 217
- V**
- validations
  - semantic input, 263–264
  - syntactic data, 260–262
  - using service repository, 483
- value chain diagram
  - case study, 499–500
  - defined, 122
  - overview of, 125–126
  - reasons to use, 131
  - service design process, 503–504
- value constraints, information modeling, 176
- variability, evaluating, 596, 617–618
- VB (Visual Basic), 4–5, 8–10
- vehicle identification number (VIN), 171
- vendors, integration using Web Services of, 576–578
- Venetian Blind, XML design pattern, 197–198
- verification, DSLs vs. GPLs, 289
- versioning
  - analyzing, 597
  - creating reference architecture, 84
  - dealing with service changes, 325–327
  - deployment and access approaches, 327–329
  - designing XML documents for, 188–189
  - developing enterprise policy, 479
  - example of, 329
  - integration, 383–384
  - overview of, 67
  - service repository and, 484
  - support in XML schemas, 189–190
- views
  - architectural, 30–31
  - software architectural, 46–48
- VIN (vehicle identification number), 171
- virtualization, 345, 386–389
- visibility
  - analyzing, 596
  - defined, 206–207
  - service interface design, 207
- Vision, Business Motivation Model, 133–134, 501
- Visual Basic (VB), 4–5, 8–10
- Visual Studio, 565–566
- W**
- Web Ontology Language (OWL), 162
- Web Services, 562
  - infrastructure of services using, 42
  - integration based on, 369–371
  - integration based on CICS, 561–565
  - integration based on vendor's, 557, 576–578
  - integration using wrappers, 374–375
  - integration with COM components, 557, 565–568
  - integration with existing J2EE applications, 570–573
  - limitations in enterprise applications, 9
  - technology independence and, 49
  - user identity conversion requests implemented, 381
- Web Services Business Process Execution Language. *See* WS-BPEL (Web Services Business Process Execution Language)
- Web Services Choreography Language (WS-CDL), 277–278, 286–290
- Web Services Description Language. *See* WSDL (Web Services Description Language)

- Web Services Remote Portlets (WSRP), 316
- Web Services Secure Exchange (WS-SX) Technical Committee, 408, 410
- Web Services, standards and specifications, 405–416
  - SAML, 411–413
  - types of, 408–409
  - WS-Federation, 409–410
  - WS-SecureConversation, 410
  - WS-Security SOAP messaging, 405–406
  - WS-SecurityPolicy and WS-Policy Framework, 410–411
  - WS-Trust, 406–408
  - XACML, 413–415
  - XML encryption, 415–416
  - XML Signature, 415
- Web-Service-based SSO, 398, 423–424
- WebSphere MQ integration, 559–561, 563
- wires, SCA, 285
- workflow
  - BU01- quote insurance, 580–582
  - BU02- process application, 583–585
  - BU03- change policy, 585–588
  - defining, 63
  - orchestration of service composition, 276–278
- workspace tier, 319, 320–321
- WorkWithDocuments operation, 556–557
- WS-BPEL (Web Services Business Process Execution Language)
  - abstract processes for, 308–309
  - defining, 278
  - orchestration with, 276, 299–301
  - reference guide for, 299
  - service composition using, 303–307
- WS-CDL (Web Services Choreography Language), 277–278, 286–290
- WSDL (Web Services Description Language)
  - avoiding SOA stovepipes, 199–200
  - CICS integration using Web Services, 562
  - fallacy of publish-find-bind triangle, 456–457
  - generating service interface with, 16
  - integration using Web Services wrappers, 375
  - useless, 370–371
  - WS-BPEL extending, 300
  - WS-Policy complementing, 410
- WS-Federation
  - defined, 408
  - origins of, 399
  - overview of, 409–410
  - WS-Trust used with, 408
- WS-Policy
  - defined, 408
  - overview of, 410–411
  - WS-SecurityPolicy as subset of, 410
- WS-Policy framework, 486–488
- WSRP (Web Services Remote Portlets), 316
- WS-SecureConversation
  - achieving confidentiality with, 401
  - confidentiality of, 405
  - defined, 408
  - overview of, 410
  - WS-Trust used with, 408
- WS-Security, defined, 408
- WS-Security SOAP messaging
  - confidentiality of, 401
  - integrity, 403
  - non-repudiation, 404
  - overview of, 405–406
  - point-to-point authentication, 426
  - WS-Federation, 408–409
  - WS-SecureConversation, 410
  - WS-Trust, 406–408
  - XML Encryption utilized by, 415–416
  - XML Signature utilized by, 415
- WS-SecurityPolicy
  - defined, 408
  - developing and registering run-time policies, 487
  - flexibility with, 436–437

- WS-SecurityPolicy (*continued*)  
integration based on vendor's Web Services, 577–578  
overview of, 410–411
- WS-SX (Web Services Secure Exchange) Technical Committee, 408, 410
- WS-Trust  
built on WS-Security SOAP messaging, 405  
confidentiality of, 401  
defined, 408  
overview of, 406–408  
WS-Federation using and extending, 409
- X**
- X.509 Certificates, WS-Security Token Profile, 406, 426
- XACML (eXtensible Access Control Markup Language)  
authorization for access control, 395  
defined, 408–409  
overview of, 413–415
- XML (Extensible Markup Language)  
integration with Web Services and, 370  
semantic interoperability and, 160–162  
WS-BPEL based on, 300  
XACML, 413–414
- XML documents  
designing for change, 188  
overview of, 181–183  
schemas, 184–185  
signing with XML Signature, 404  
types in schemas, 185–187  
variations in schemas, 187–188  
versioning support in schemas, 189–190
- XML Encryption standard, 401, 409, 415–416
- XML namespaces, 189
- XML parsers, 189
- XML patterns, 190–198  
derivation by extension, 193–194  
derivation by restriction, 194  
derivation using abstract classes, 192–193  
disallowing derivations, 195–198  
overview of, 190–192
- XML schemas  
avoiding SOA stovepipes, 199–200  
overview of, 184–185  
as semantic technology, 162  
types in, 185–187  
variations in, 187–188  
versioning support in, 189–190
- XML Signature  
defined, 409  
overview of, 415  
point-to-point authentication using, 426  
providing integrity, 403  
providing non-repudiation, 404
- XML-DSIG. *See* XML Signature
- XML-SIG. *See* XML Signature