absolute isolation, 309, 317
abstract classes (OOAD), 461
designing service-oriented classes, 474
Abstract Syntax Notation 1 (ASN.1), 128
abstraction (OOAD), 463. See also Service Abstraction (principle)
access control levels, 232-234
accessor methods (OOAD), 454
active state (state management), 335
aggregates of services. See service compositions
aggregation (OOAD), 471-472
agile development, 87, 521
organizational agility versus, 63
service-orientation and, 87
Service Reusability design risks, 287
agility. See organizational agility
agnostic capability candidates, 523
agnostic service references, 63
agnostic services, 62, 82, 91, 407
reusable services versus, 268-269
service contracts, 144
Service Reusability, 268-269
agnostic solution logic, increasing, 82
alignment of business and technology. See business and technology domain alignment in service-oriented computing
analysis phase, measuring service reusability in, 265-266
analysis scope, defining, 522
AOP (aspect-oriented programming), as an influence of service-orientation, 99, 448
API (application programming interface), 48, 128, 174, 177, 213, 313
functional abstraction, 221
service contracts and, 129
application architectures, 95-96
application programming interface. See API
application services. See utility services
application-specific solution logic, reducing, 82-83
applications
composite, 91-92
service compositions versus, 91-92
service-orientation and, 91-92
technology architectures, 95-96
architects. See enterprise architects (role)
architecture. See also SOA (service-oriented architecture)
application, 95-96
client-server, 128, 165
state management, 328
defining, 520
distributed, 128, 166
state management, 329, 331
enterprise, 80, 95-96
integration, 81, 92-96, 182-184
mainframe, 166
point-to-point, 80, 405-406
service composition, 96
Service Statelessness design risks, 349-350
of Web services, 48-49, 166
ASN.1 (Abstract Syntax Notation 1), 128
aspect-oriented programming. See AOP assertions. See policy assertions
association (OOAD)
comparison of object-orientation and service-orientation, 469-470
designing service-oriented classes, 474
attachments (SOAP), 334
attributes (objects), explained, 454
attributes (OOAD), 473
auditors. See enterprise design standards custodians (role)
auto-generation (of service contracts), 175, 178
autonomy. See also Service Autonomy (principle)
composition autonomy, 430
data models and, 308-310
databases and, 308-310
governance and, 298-299
service compositions and, 298, 314

B
base classes (OOAD), 461
benefits of service-oriented computing. See service-oriented computing, goals and benefits
best practices
architecture dependency, 350
building Web services, 151
controlled access, 234
discoverability meta information, 382
Domain Inventory design pattern, 275
encapsulated legacy environments, 318
example of, 34
explained, 34-35
measuring consumer coupling, 192
service composition performance limitations, 437
service contract design risks, 150
for service-orientation, 87
bidirectional coupling, 165
black box concept, 213, 227
books, related, 4-5
Web site, 16
bottom-up processes, 518-519
BPM (business process management), as an influence of service-orientation, 98, 448
bridging products, 142
business agility. See organizational agility
business analysts, 522
discoverability meta information and, 377
role of, 53
business and technology domain alignment in service-oriented computing, 60-61
Index

business data (state management), 338
business entity services. See entity services
business logic. See core service logic in Web sites
business models. See enterprise business models
business process definition, explained, 397
business process instance, explained, 397
business process management. See BPM
business process services. See orchestrated task services; task services
business requirements fulfillment, as goal of object-orientation, 450-451
business service candidates, 377
business services. See entity services; task services

C

candidates. See service candidates
capabilities
granularity and, 116
operations and methods versus, 115
service compositions, 399-400
services and, 69-70
capability candidates. See service capability candidates
capability granularity, 486
explained, 116
Service Composability and, 428
service contracts, 143
Service Loose Coupling principle and, 195-196
Service Reusability and, 277
Capability Name (service profile field), 481
capability profiles, structure of, 481-482
case study
background, 20-22, 66, 100-101, 119-121
business process description, 119-121
conclusion of, 514-515
coupling in, 202-209
preliminary planning, 101
service abstraction levels, 244-252
Service Autonomy in, 319-323
Service Composability in, 439-441
Service Discoverability in, 382-386
Service Reusability in, 288-292
Service Statelessness in, 351-359
services in, 154
Standardized Service Contract principle example, 154-161
style, 20
centralization
Contract Centralization design pattern, 185, 195, 473, 530
example of, 216-217
Logic Centralization and,
272-273
measuring consumer coupling,
191-192
standardized coupling and, 185
technology coupling, 189-190
Logic Centralization design pattern, 185, 465, 468, 531
Contract Centralization and,
272-273
difficulty in achieving, 274-275
as enterprise design
standard, 272
explained, 271
standardized coupling and, 185
Web services and, 274
of policy assertions, 138-139
Schema Centralization design pattern, 135-137, 531
characteristics. See design characteristics
chorded circle symbol, explained, 13, 15-16
classes (OOAD)
  compared to service contracts, 453
  service-oriented classes, 472-474
client-server architectures, 165, 128
  state management, 328
course-grained design. See granularity
code examples
  capability expressed in IDL, 129
  capability expressed in WSDL, 129
  constraint granularity, 117
  fine-grained XML schema simple type, 143
  skeleton (coarse- and fine-grained) operation definitions, 143
  skeleton WSDL definition for coarse-grained service, 142
  SOAP and WS-Addressing headers for state management, 337
  standardized and non-standardized WSDL message definitions, 133
  UDDI discoveryURL construct, 372
  WS-BPEL composition logic, 431
  WS-Coordination headers for state management, 338
  WS-MetadataExchange and WS-Addressing, 373
cohesion
  comparison of object-orientation and service-orientation, 467
  service granularity and, 467
collective composability, explained, 400-401
color, in symbols, 13
commercial product design, 62, 276
  abstraction and, 214
  coupling and, 166
gold-plating versus, 267
meta abstraction types in, 227
  measuring service reusability, 262, 264-265
  risks associated with, 286-287
communications quality, 365
communications specialists. See technical communications specialists (role)
complete reusability, 266, 487
complex compositions. See complex service compositions
  complex service activities, 402
  complex service compositions, 406-407, 487
  characteristics of, 410-411
  preparation for, 411
  service inventory evolution, 407, 409-410
complexity, in traditional solution delivery, 80
components, coupling and, 176-177
composability. See Service Composability (principle)
composition (OOAD), 470-471. See also service compositions; Service Composability (principle)
composition autonomy, 430
  Service Composability and, 430
composition candidates. See service composition candidates
composition controller capabilities, 394, 400
composition controllers, 435, 487
  explained, 398-401
  service consumers as, 404
Index

composition initiators, 487
 explained, 403-405
 service consumers as, 404
 composition instances, 397
 composition member capabilities, 393, 400
 Composition Member Capabilities (service profile field), 481
 composition members, 487
design of. See Service Composability (principle)
 explained, 398-401
 Web service region of influence for, 395
 Composition Role (service profile field), 481
 composition sub-controllers, 487
 concise contract abstraction, 232, 487
 conflict symbol, 13
 constraint granularity, 486
 explained, 117-118
 Service Abstraction and, 239
 Service Composability and, 428
 service contracts, 143
 Service Loose Coupling principle and, 195-196
 Service Reusability and, 278
 consumer coupling
 measuring, 191-192
 Service Abstraction and, 192
 Service Composability and, 191
 service consumers, 48-49
 as composition initiators and controllers, 404
 coupling and, 167
 coupling types, 181-192
 policy dependencies, 138
 consumer-specific functional coupling, 180
 consumer-to-contract coupling, 185-191, 473, 486
 risks with, 214
 Web services and, 186
 consumer-to-implementation coupling, 182, 184, 486
 integration architectures and, 182-184
 containers, objects as, 458
 content abstraction, 246
 context data (state management), 337-338
 context rules (state management), 337
 Contract Centralization design pattern, 185, 195, 473, 530
 example of, 216-217
 Logic Centralization and, 272-273
 measuring consumer coupling, 191-192
 standardized coupling and, 185
 technology coupling, 189-190
 contract content abstraction levels, 231-232
 Contract Denormalization design pattern, 242, 305, 312, 530
 service contract autonomy and, 304-305
 contract first design, 53, 131, 173, 194
 contract-to-functional coupling, 180, 486
 indirect consumer coupling and, 188
 contract-to-implementation coupling, 177-179, 486
 examples of, 177
 indirect consumer coupling and, 189
 service composability, 200
contract-to-logic coupling, 174-175, 486
policies and, 179
Service Composability and, 199
contract-to-technology coupling,
176-177, 486
direct consumer coupling and, 188
Service Composability and, 199
contracts. See service contracts
controlled access (access control level),
233-234, 487
controller capabilities, 400
controllers. See composition controllers
core service logic in Web services, 48
coupling. See also Service Loose
Coupling (principle)
architectural, 168
auto-generation and, 175
in case study, 202-209
in client-service architectures, 165
commercial product design
and, 166
compared to dependency, 165
data models and, 175
database tables and, 175
design principles, relationship
with, 197-200
design risks, 200
logic-to-contract coupling,
200-201
performance problems, 201-202
design-time autonomy and, 181,
315-316
in distributed architectures, 166
explained, 164-165
integration architectures and,
182-184
mainframe and, 166
multi-consumer coupling
requirements (Service
Abstraction principle), 242
negative types, 193, 195
in object-orientation, 166
origins of, 165-166
performance, 202
policies and, 179
positive types, 193, 195
proprietary components and,
176-177
risks with, 214
Service Composability and, 191
service consumer coupling types,
181-182
consumer-to-contract coupling,
185-191
customer-to-implementation
coupling, 182, 184
Contract Centralization design
pattern, 185
measuring consumer coupling,
191-192
service contract coupling types,
169-173
contract-to-functional
coupling, 180
contract-to-implementation
coupling, 177-179
contract-to-logic coupling,
174-175
contract-to-technology coupling,
176-177
logic-to-contract coupling,
173-174
service granularity and, 195-196
service models and, 196-197
service-orientation and, 193-195
symbols for, 165
Web services and, 166
coupling quality, 146
cross-cutting functions, 313, 347
CRUD, 44, 464
Index

cultural issues, Service Reusability design risks, 281-283
Custodian (service profile field), 482
Cutit Saws case study. See case study

D

data granularity, 486
explained, 116
Service Composability and, 428
service contracts, 143
Service Loose Coupling principle and, 195-196
Service Reusability and, 278
data models
autonomy and, 308-310
contract-to-implementation coupling and, 177-178
coupling and, 175
data granularity and, 116
example of coupling, 206
global, 136
logical, 52
service contracts and, 134-137
standardization, 50, 89, 134-137
data representation standardization, 134-137
case study, 155
data transformation, avoiding, 140-142
sample design standards, 155
data transformation
avoidance, 135-136, 140-142
design standards and, 135-136
performance issues, 140
problems, 140
standardization and, 140-142
Standardized Service Contract principle and, 135-136, 140-142
databases
autonomy and, 308-310
contract-to-implementation coupling and, 177-178
coupling and, 175
for state management, 329, 331, 339-343
dedicated controllers, 487
deferral. See state deferral
delegation (OOAD), 468-469. See also state delegation
delivery processes. See processes
delivery strategies. See processes
denormalization. See also normalization
service contracts and, 301-305
dependency, coupling compared to, 165
design characteristics
example of, 27
explained, 27-28
implementation of, 111-114
importance of, 69
list of, 81
loose coupling, 166
regulation of, 111-114
design framework, 35-36
design granularity. See granularity
design paradigm
example of, 29
explained, 29-30
relationships with design framework, 36
service-orientation as, 70-71
design pattern language
example of, 32
explained, 31-32
design patterns

Contract Centralization design pattern, 185, 195, 242, 473, 530
example of, 216-217
Logic Centralization and, 272-273
measuring consumer coupling, 191-192
standardized coupling and, 185
technology coupling, 189-190
Contract Denormalization, 242, 305, 312
service contract autonomy and, 304-305

Domain Inventory, 136, 275
domain inventory example of, 31
domain inventory explained, 30-31
domain inventory how they are referenced, 111
Logic Centralization, 185, 465, 468
Contract Centralization and, 272-273
difficulty in achieving, 274-275
as enterprise design standard, 272
explained, 271
Web services and, 274

referred in design principles, 530
relationships with design framework, 36

Schema Centralization, 135-137

Service Normalization, 272, 305, 465
service contract autonomy and, 302-304
design phase (service composition), 413
assessments, 413, 415
design principles
application levels, vocabularies for, 487-488
best practices versus, 34

business and technology alignment in, 502-503
comparing to object-oriented design principles, 457-472
design pattern references, 111, 530
design standards and, 33, 107-108
documentation for, 109-110
documentation for, example of, 28
explained in abstract, 28-29
extent of abstract example of, 28
federation in, 501
in formal service design processes, 106-107
granularity, types of, 115-118
guidelines for working with, 104-110, 115-121
implementation mediums and, 114-115
implementation of design characteristics, 111-114
interoperability and, 74-75
intrinsic interoperability in, 498, 500
list of, 71-73
mapping to strategic goals, 498-509
organizational agility in, 505, 507
principle profiles, explained, 109-110
reduced IT burden in, 509
regulation of design characteristics, 111-114
ROI in, 504
Service Abstraction, relationship with, 239-241. See also Service Abstraction (principle)
Service Autonomy, relationship with, 314-317. See also Service Autonomy (principle)
Index

Service Composability, relationship with, 432-436. See also Service Composability (principle)
service contracts. See service contracts
Service Coupling (principle), relationship with, 197-200
Service Discoverability, relationship with, 378-380. See also Service Discoverability (principle)
Service Reusability (principle), relationship with, 278, 280-281
Service Statelessness, relationship with, 347-349. See also Service Statelessness (principle)
in service-oriented analysis, 105-106
service-oriented computing elements, relationship with, 41
SOA goals and benefits, relationship with, 498-499
standard structure, 109-110
standardization of service contracts, relationship with, 144-148
vendor diversification in, 501-502
vocabularies for, 486-487
design standards
data representation design standard samples, 155
design principles and, 107-108
example of, 33
explained, 32-33
functional expression design standard samples, 155
granularity and, 144
importance of, 86
industry standards versus, 34
level required, 89
naming conventions, 147
in service-orientation, 86
Standardized Service Contract principle and, 132
design taxonomy, 35
design-time autonomy, 486
coupling and, 315-316
explained, 298-299
logic-to-contract coupling and, 181
service contracts and, 301-305
design-time discovery, 371-373, 486
design-time isolation, 309
designated controllers, explained, 400
detailed contract abstraction level, 231, 487
development tool deficiencies, 151-152
direct consumer coupling example of, 188
indirect consumer coupling versus, 186, 188-189
discoverability, explained, 364. See also Service Discoverability (principle)
discovery. See also Service Discoverability (principle)
explained, 364-366
meta information and, 362
origins of, 367-368
processes, 363-367
of resources, 362-368
types of, 371-373
distributed architectures, 128, 166
state management, 329, 331
dLL (dynamic link library), 390
document-centric messages, 117
Domain Inventory design pattern, 136, 275, 531
don’t repeat yourself. See DRY (OOAD)
DRY (OOAD), 465-466
dynamic link library. See DLL
EAI, 213, 448
as an influence of service-orientation, 98-99, 448
encapsulation
of legacy logic, 318
Service Abstraction versus, 235
service encapsulation, 235-237
encapsulation (OOAD), 458
Endpoint References, 345
to enterprise application integration.
See EAI
to enterprise architects (role), 494-495
to enterprise architectures, 80, 95-96
to enterprise business models,
defining, 520
to enterprise design standards custodians (role), 495
to entity schemas, 136
to entity services
coupling and, 196
design processes, 526
example of, 44
explained, 44
Service Abstraction principle, 239
Service Autonomy and, 312-313
service contracts, 144
Service Statelessness and, 346
to entity-centric business services. See to entity services
to entity-centric schemas, 137
to errata, 16
to event-driven, 48
to examples. See (case study; code examples; For Example sections
to extends attribute, 460
to extensibility, as goal of object-orientation, 450-451

F
façade classes (OOAD), designing service-oriented classes, 474
to federated service architecture, 59
federation
in service-oriented computing, 58-59
with services, 58
Web services and, 59
fine-grained design. See granularity
first-generation Web services platform, 47. See also Web services
flexibility, as goal of object-orientation, 450, 452
For Example sections
to composition initiators, 404-405
to contract-to-implementation coupling, 179
to contract-to-logic coupling, 175
to contract-to-technology coupling, 177
to design standards, 108
to formal service design processes, 107
to logic-to-contract coupling, 174
to messaging, 344
Service Abstraction principle, 216-217
to service contract autonomy, 303
to service modeling process, 106
Service Reusability, 284-285
XML schema standardization, 137
fully deferred state management, measuring service statelessness, 342-343
functional abstraction, 221-222, 225, 486
example of, 246
functional context, 70, 312, 468
service granularity and, 116
Index

functional coupling. See contract-to-functional coupling
functional expression
standardization, 155
functional isolation, 308
functional meta data, 374, 486
example of, 383-386
functional scope, Service Autonomy
design risks, 317
functional service expression,
standardization of, 133-134
case study, 155
fundamental concepts, comparison of
object-orientation and service-orientation, 453-454, 456-457

G
generalization (OOAD), 461-462
global data models, 136
glossary Web site, 16, 533
goals
comparison of object-orientation
and service-orientation, 449-452
mapping to design principles, 498-509
goals of service-oriented computing. See service-oriented computing,
goals and benefits
gold-plating, 267
governance
autonomy and, 298-299, 316
design-time autonomy and,
298-299
pure autonomy, 308
reuse and, 316
Service Composability design
risks, 438
Service Reusability design risks, 283-285
of service-orientation, 88
governance phase (service
composition), 413
assessment, 417, 419
granularity. See also capability
granularity; constraint granularity;
data granularity; service granularity
design standards and, 144
levels, 118
types of, 115-118
Guidelines for Policy Assertion Authors
(W3C), 493

H
hardware accelerators, 334
has-a relationships (OOAD),
469-471, 474
hidden compositions, 402, 434
hiding information. See Service
Abstraction (principle)
high statelessness, 342-343
history. See origins

I
IDL (Interface Definition
Language), 128
implementation coupling, example of,
206-207
implementation mediums, design
principles and, 114-115
implementation phase, measuring
service reusability in, 267
implementation principles, 111-114
implementation requirement, service
contracts, 131
increased intrinsic interoperability, 75
indirect consumer coupling
direct consumer coupling versus,
186, 188-189
example of, 188-189, 207
industry standards, design standards versus, 34. See also Web services
information architecture models, 52
information hiding. See Service Abstraction (principle)
infrastructure services. See utility services
inheritance (OOAD), 166
    comparison of object-orientation and service-orientation, 459-460
designing service-oriented classes, 473
    service granularity and, 473
Input/Output (service profile field), 481
    integration of architectures, 81
customer-to-implementation coupling, 182-184
coupling and, 182-184
EAI (enterprise application integration), 98-99
    service compositions and, 92-94
    service-orientation and, 84, 92-94
    in traditional solution delivery, 80-81
integration architectures, 95-96
Interface Definition Language. See IDL
interface element, 456
interfaces (OOAD)
    compared to service contracts, 456-457
    compared to WSDL portType and interface elements, 456
designing service-oriented classes, 473
    measuring service statelessness, 342
interoperability
    of services, 84
    service-orientation and, 74-75, 84
    in service-oriented computing, 56-57
interpretability. See also Service Discoverability (principle)
    defined, 365
    explained, 365
interpretation process, 364-367
    explained, 365
intrinsic interoperability. See interoperability
inventory analysis, 520-521, 523
is-a relationships (OOAD), 459
is-a-kind-of relationships (OOAD), 461
isolation
    levels of, 308-310
    partially isolated services, 306-308
    of services, 308-310
IT roles. See organizational roles

J–K
JDBC, 166
Keywords (service profile field), 481

L
LDAP directories, 367
legacy systems
    effect on, 523
    mainframe architectures, 166
    Service Autonomy design risks, 318
    service encapsulation, 236
lifecycle phases of service
    composability, 412-413
logic abstraction. See programmatic logic abstraction
Logic Centralization design pattern, 185, 465, 468, 531  
Contract Centralization and, 272-273  
difficulty in achieving, 274-275  
as enterprise design standard, 272  
explained, 271  
standardized coupling and, 185  
Web services and, 274  

Logic Description (service profile field), 481  
logic-to-contract coupling, 173-174, 486  
design-time autonomy and, 181  
external limitations, 200-201  
Web services and, 201  
logic-to-implementation coupling, 178  
logical data models, 52  
loose coupling. See Service Loose Coupling (principle)  
low-to-no statelessness, 340  

M  
mainframe architectures, 166  
measuring  
consumer coupling, 191-192  
Service Abstraction, 231  
access control abstraction levels, 232-234  
contract content abstraction levels, 231-232  
quality of service meta information, 234  
Service Autonomy, 300-301  
mixed autonomy, 310  
pure autonomy, 308-310  
Service contract autonomy, 301-305  
Service logic autonomy, 306-308  
shared autonomy, 305-306  

Service Composability, 412  
checklists, 419-420, 426-427  
design phase assessment, 413, 415  
governance phase assessment, 417, 419  
lifecycle phases, 412-413  
runtime phase assessment, 415, 417  

Service Discoverability  
baseline measures checklist, 375-376  
custom measures, 376  

Service Reusability, 262-263  
in analysis/design phase, 265-266  
commercial design approach, 262, 264-265  
gold-plating, 267  
in implementation phase, 267  

Service Statelessness, 339  
fully deferred state management, 342-343  
internally deferred state management, 342  
non-deferred state management, 340  
partially deferred memory, 340-341  
partially deferred state management, 341-342  

message correlation, 337  
message processing logic for Web services, 48  
messages. See also SOAP  
comparison of object-orientation and service-orientation, 454-456  
data granularity and, 116  
document-centric, 117  
RPC-style, 117  
as state deferral option, 343-344
meta abstraction types, 218-219
  in commercial software, 227
  in custom-developed software, 228-229
  functional abstraction, 221-222
  in open source software, 227-228
  programmatic logic abstraction, 222-223
  quality of service abstraction, 224
  technology information abstraction, 219-221
  Web service design and, 225-226
  in Web services, 229-230
meta information types. See Service Discoverability (principle)
methods (objects), explained, 454
mixed autonomy, 310, 313
mixed detailed contract abstraction level, 232, 487
moderate statelessness, 341-342
modularization of policy assertions, 138-139
monolithic executables, 390
multi-consumer coupling requirements (Service Abstraction principle), 242
multi-purpose logic, 268
multi-purpose programs, 255-256
multi-purpose services, 468
non-technical service contracts, 152-153. See also SLA
  Service Abstraction and, 237-238
normalization
  Contract Denormalization design pattern, 305, 312, 530
  service contract autonomy and, 304-305
  entity services, 313
  service contracts and, 301-305
  Service Normalization design pattern, 272, 305, 465, 531
  service contract autonomy and, 302-304
  of services, 65, 83
  utility services, 313
notification service for updates to
  Prentice Hall Service-Oriented Computing Series from Thomas Erl
  books, 17, 533
object-orientation, 129
  abstract classes, 461
designing service-oriented classes, 474
  abstraction, 213, 463. See also
  Service Abstraction (principle)
  accessor methods, 454
  aggregation, 471-472
  association
comparison of object-orientation and service-orientation, 469-470
designing service-oriented classes, 474
  attributes, 473
  base classes, 461
naming conventions. See vocabularies
negative types of coupling, 193, 195
nested policy assertions, 138
.NET, 177, 216-217
no access (access control level), 234, 487
non-agnostic capability candidates, 523
non-deferred state management, 340

N
naming conventions. See vocabularies
negative types of coupling, 193, 195
nested policy assertions, 138
.NET, 177, 216-217
no access (access control level), 234, 487
non-agnostic capability candidates, 523
non-deferred state management, 340
Index

classes
  compared to service contracts, 453
  service-oriented classes, 472-474
composition, 470-471. See also service compositions; Service Composability (principle)
coupling, 166
delegation, 468-469. See also state delegation
as design paradigm, 30
DRY, 465-466
encapsulation, 458
façade classes, designing service-oriented classes, 474
generalization, 461-462
has-a relationships, 469-471, 474
as influence of Service Composability, 391
as influence of service-orientation, 97
inheritance, 166
  comparison of object-orientation and service-orientation, 459-460
designing service-oriented classes, 473
service granularity and, 473
interfaces
  compared to service contracts, 456-457
  compared to WSDL portType and interface elements, 456
designing service-oriented classes, 473
measuring service statelessness, 342
is-a relationships, 459
is-a-kind-of relationships, 461
OCP, 465
polymorphism, 463-464
reuse and, 257
RPC, 448
service-orientation compared, 97, 446-475
commum goals, 449-452
design principles, 457-472
fundamental concepts, 453-457
specialization, 461-462
SRP, 466-468
sub-classes, 459, 461, 463
super-classes, 459
uses-a relationships, 469, 471, 474
object-oriented design principles,
  compared to service-orientation design principles, 457-458, 460-471
objects
  compared to services, 453
  as containers, 458
OCP (OOAD), 465
ODBC, 166
ontologies, 52
OOAD (object-oriented analysis and design). See object-orientation
open access (access control level), 233, 487
open source software, meta abstraction types in, 227-228
open-closed principle. See OCP
optimized contract abstraction level, 232, 487
orchestrated task services
  coupling and, 197
defined, 45
  Service Abstraction principle, 239
  Service Autonomy and, 313-314
Service Composability and, 430, 432
Service Statelessness and, 347
orchestration. See orchestrated task services; WS-BPEL
orchestration services. See orchestrated task services
organizational agility
agile development versus, 63
project delivery timelines and, 64
combinability and, 64
responsiveness and, 63
Service Abstraction principle
support for, 506
service compositions and, 64
Service Loose Coupling principle
support for, 506
Service Reusability Coupling principle
support for, 64, 506
service-orientation and, 63
in service-oriented computing, 63-64
organizational culture. See cultural issues
organizational roles, 488-490
enterprise architects, 494-495
teachers, 496
enterprise design standards custodians, 495
design custodians, 495
policy custodians, 493
schema custodians, 492
service analysts, 491
service architects, 491
service custodians, 492
service registry custodians, 493-494
teachers, 494
technical communications specialists, 494
origins
of autonomy, 295
of composition, 390-392
of coupling, 165-166
of discovery, 367-368
of information hiding, 213
of reuse, 257-258
of service-orientation, 96-99
AOP (aspect-oriented programming), 99
BPM (business process management), 98
EAI (enterprise application integration), 98-99
object-orientation, 97
Web services, 98
of service contracts, 127-129
of state management, 328-331
overestimating service usage requirements, 318
P
paradigm. See design paradigm
parameters in policy assertions, 138
parent process coupling, 180
partially deferred memory, 340-341
partially deferred state management, 341-342
partially isolated services, 306-308
passive state (state management), 335
pattern languages. See design pattern languages
patterns. See design patterns
performance
data transformation, 140
schema coupling and, 202
Service Composability design risks, 437-438
service loose coupling, 201-202
state management and, 334
Plain Old XML. See POX
planned reuse, measures of, 265-266
Index

point-to-point data exchanges, explained, 80, 405-406
policies, 48, 137-139, 274, 493
  centralization and, 138
  contract-to-logic coupling, 179
  editors, 152
  processors, 138
  Service Abstraction and, 238
  service consumer dependencies and, 138
  service profiles and, 483
  structural standards, 139
policy alternatives, 378
policy assertions, 146, 493
  centralization, 138-139
  modularization, 138-139
  nested policy assertions, 138
  parameters, 138
  proprietary vocabularies for discoverability, 378
  Service Discoverability and, 378
  structural design, 139
  structural standards and, 139
  vocabularies for, 137-138
policy custodians (role), 493
policy parameters, 378
policy vocabularies, 493
polymorphism (OOAD), 463-464
portType element, 456
positive types of coupling, 193, 195
post-implementation application of service discoverability, 381
poster Web site, 16, 534
POX (Plain Old XML), 50
Prentice Hall Service-Oriented Computing Series from Thomas Erl, 4, 111, 284, 495, 531
  Web site, 16, 533
primitive compositions, 406, 487
primitive service activities, 402, 405
principle profiles
  explained, 109-110
  Service Abstraction, 214-217
  Service Autonomy, 296-297
  Service Composability, 392, 395-396
  Service Discoverability, 368, 370
  Service Loose Coupling, 167, 169
  service profiles versus, 110
  Service Reusability, 259-261
  Service Statelessness, 331-332, 334
  Standardized Service Contract, 130-132
principles. See design principles
privacy concerns, Service Abstraction principle, 243
process services. See orchestrated task services
process-specific services, service contracts for, 144
processes
  bottom-up, 518-519
  choosing, 521-522
  discovery, 363-367
  interpretation, 364-367
  inventory analysis cycle, 520-521
  service delivery, 518, 521-528
  service modeling, 105-106, 523
  service-oriented analysis, 105-106, 521
  service-oriented design, 106-107
  SOA delivery, 518, 521-528
  top-down, 518-519
productivity, as goal of object-orientation, 450, 452
profiles. See principle profiles; service profiles
programmatic logic abstraction, 222-223, 226, 486
proprietary assertion vocabularies, 378
proprietary vocabularies, 137-138
proxies, 128
pure autonomy, 308-310, 317, 488
Purpose Description (service profile field), 481
QoS Requirements (service profile field), 481
quality of service abstraction, 224, 226, 486
quality of service meta information, 374, 486
abstraction levels and, 234
example of, 386
reduced IT burden, as supported by Service Composability principle, 509
reduced statefulness, 340-341
redundancy
avoidance of, 64, 465-466
reducing, 83
in silo-based applications, 78
in traditional solution delivery, 78-79
registries. See service registries
regulatory presence, 241
regulatory principles, 111-114
reliability, 317
Service Reusability design risks, 286
repository versus registry, 367
REST (Representational State Transfer), 50
return on investment. See ROI
reusability, 69. See also Service Reusability (principle)
as goal of object-orientation, 450, 452
level required, 90
reuse versus, 256
reusable components (Standardized Service Contract principle), 129
reuse, 62-63, 69, 82, 90. See also Service Reusability (principle)
explained in abstract, 254-256
governance rigidity and, 438
origins of, 257-258
reusability versus, 256
traditional approaches, 258
traditional problems with, 257-258
Web services and, 258
risks
with consumer-to-contract coupling, 214
of gold-plating, 267
Service Abstraction design, 242
human misjudgment, 242-243
multi-consumer coupling requirements, 242
security and privacy concerns, 243
Service Autonomy design
functional scope, 317
overestimating service usage requirements, 318
wrapper services, 318
Service Composability design
governance rigidity, 438
performance limitations, 437-438
single points of failure, 437
Index

Service Contract design, 149
  development tool deficiencies, 151-152
  technology dependencies, 150
  versioning, 149-150
Service Discoverability design
  communication limitations, 381-382
  post-implementation
  application, 381
Service Loose Coupling design, 200
  logic-to-contract coupling, 200-201
  performance problems, 201-202
Service Reusability design, 281
  agile delivery, 287
  commercial design, 286-287
  governance structure, 283-285
  organizational culture, 281-283
  reliability, 286
  security, 286
Service Statelessness design
  architecture dependency, 349-350
  runtime performance, 350
  underestimating effort
  requirements, 350
  robustness, as goal of object-orientation, 450-451
ROI (return on investment)
  Service Composability principle
  support for, 505
  Service Discoverability principle
  support for, 505
  Service Statelessness principle
  support for, 505
in service-oriented computing, 61-62
roles. See organizational roles
RPC, 150, 448, 455
RPC-style messages, 117
runtime autonomy, 486
  explained, 298
  normalization design patterns, 305
  service contracts and, 301-305
runtime discovery, 371-373, 486
runtime performance (Service Statelessness design risks), 350
S
  scalability, 326, 333, 340, 348
Schema Centralization design pattern, 135-137, 531
schema custodians (role), 492
scope
  of analysis, defining, 522
  comparison of object-orientation and service-orientation, 447
second-generation Web services platform, 47. See also Web services
security
  Service Abstraction principle, 243
  Service Reusability design risks, 286
separation of concerns, 70
  in relation to service compositions, 390
Service Abstraction (principle), 72, 212-251, 402
  application level terminology, 487
  associated terminology, 486
  in case study, 244-252
  commercial product design
  and, 214
compared to abstraction
(OOAD), 463
considerations when designing
service-oriented classes, 473
constraint granularity and, 239
customer coupling and, 192
contribution to realizing
organizational agility, 506
design principles, relationship
with, 239-241
design risks, 242
  human misjudgment, 242-243
  multi-consumer coupling
requirements, 242
  security and privacy
  concerns, 243
effect on other design principles,
239-241
encapsulation versus, 235-237
explained, 212
goals, 215
impact on composition design
process, 418
implementation requirements, 216
interoperability and, 74
measuring, 231
  access control abstraction levels,
  232-234
  contract content abstraction
  levels, 231-232
  quality of service meta
  information, 234
meta abstraction types, 218-219
in commercial software, 227
in custom-developed software,
228-229
functional abstraction, 221-222
in open source software, 227-228
programmatic logic abstraction,
222-223
quality of service
abstraction, 224
technology information
abstraction, 219-221
Web service design and, 225-226
in Web services, 229-230
non-technical contract documents
and, 237-238
origins of, 213
policies and, 238
policy assertions, 238
principle profile, 214-217
Service Autonomy and, 316
Service Composability and, 241,
433-435
Service Discoverability and,
241, 379
service granularity and, 238-239
Service Loose Coupling and, 114,
198, 241
service models and, 239
Service Reusability and, 241, 279
Standardized Service Contract
principle and, 146, 240
Web services and, 50
WS-Policy definitions, 238
service activities, explained,
402-403, 487
service adapters, 142, 174, 213
service agents, 114
  in message processing logic, 48
service analysts (role), 491
service architects (role), 491
Service Autonomy (principle), 72, 276,
294-323
  application level terminology, 488
  associated terminology, 486
  in case study, 319-323
composition autonomy and, 430
considerations when designing service-oriented classes, 473
coupling and, 178
design principles, relationship with, 314-317
design risks
  functional scope, 317
  overestimating service usage requirements, 318
  wrapper services, 318
design-time autonomy, explained, 298-299
effect on other design principles, 314-317
explained, 294-295
interoperability and, 74
measuring, 300-301
  mixed autonomy, 310
  pure autonomy, 308-310
  service contract autonomy, 301-305
  service logic autonomy, 306-308
  shared autonomy, 305-306
origins of, 295
principle profile, 296-297
runtime autonomy, explained, 298
scalability, 261
Service Abstraction and, 316
Service Composability and, 317, 435-436
service contracts, 301-305
service granularity and, 311-312
Service Loose Coupling and, 178, 199, 315-316
service models and, 105, 311-314, 525
Service Reusability and, 280, 316
Service Statelessness and, 316, 348
service-oriented analysis processes and, 105
Standardized Service Contract and, 301-305, 315
service candidates, 269, 276. See also service modeling explained, 52
Service Discoverability and, 377
service inventory blueprint definition, 520
service modeling and, 52
service-oriented design and, 53
services versus, 52
service capabilities
  composition design support, assessment for, 422
  composition governance support, assessment for, 426
  composition runtime support, assessment for, 424
  explained, 115
  granularity and, 116
  operations and methods versus, 115
service capability candidates, 523, 525. See also service candidates
service catalogs, service profiles and, 483
Service Composability (principle), 73, 388-441. See also composition (OOAD)
  associated terminology, 487
  in case study, 439-441
  composition autonomy and, 430
  composition controllers, explained, 398-401
  composition initiators, explained, 403-405
  composition members, explained, 398-401
  considerations when designing service-oriented classes, 473-474
consumer coupling and, 191
contract-to-implementation
coupling and, 200
contract-to-logic coupling and, 199
contract-to-technology coupling
and, 199
contribution to realizing reduced
IT burden, 509
contribution to realizing ROI, 505
design principles, relationship
with, 432-436
design risks
governance rigidity, 438
performance limitations, 437-438
single points of failure, 437
effect on other design principles,
432-436
explained, 388
interoperability and, 75
measuring, 412
checklists, 419-420, 426-427
design phase assessment,
413, 415
governance phase assessment,
417, 419
lifecycle phases, 412-413
runtime phase assessment,
415, 417
orchestration and, 430, 432
point-to-point data exchanges,
explained, 405-406
principle profile, 392, 395-396
Service Abstraction and, 241,
433-435
service activities, explained,
402-403
Service Autonomy and, 317,
435-436
service composition instances,
explained, 397
service compositions
capabilities, 399-400
explained, 397
Service Discoverability and,
380, 436
service granularity and, 427-428
Service Loose Coupling and,
199-200, 433
service models and, 428-430
Service Reusability and, 280, 435
Service Statelessness and, 436
Standardized Service Contract
and, 148, 432
Web service region of influence,
395-396
Web services and, 50, 401
service composition candidates, 523
service composition instances,
explained, 397
service composition references, 63
service compositions, 82
agnostic services, 62
applications versus, 91-92
architecture of, 95-96
autonomy and, 298, 314
capabilities, 399-400
compared to applications and
integrated applications, 94-95
complex service compositions, 407
capabilities of, 410-411
preparation for, 411
service inventory evolution, 407,
409-410
composition autonomy, 430
consumer coupling and, 191
defined, 39
design assessment, 413
evolutionary cycles, 412-413
design phase, 413
governance phase, 413
runtime phase, 413
explained, 39-40, 94-95, 388-390, 397
governance assessment, 417
governance considerations, 438
hidden, 434
implementation, 42
integrated applications versus, 92-94
naming, 96
origins of, 390-392
as related to service inventories, 407
relationship with service-oriented computing elements, 40
roles
composition controllers, 398-399
composition initiators, 403-404
composition members, 398-399
designated controllers, 400
examples of, 404-405
runtime assessment, 415, 417
scope of, 405-406
service contracts and, 148
state management and, 340
types of, 406
service consumers, 48-49
as composition initiators and controllers, 404
coupling and, 167
coupling types, 181-182
customer-to-contract coupling, 185-191
customer-to-implementation coupling, 182, 184

Contract Centralization design pattern, 185
measuring consumer coupling, 191-192
policy dependencies, 138
service contract autonomy, 301-305, 488
service contracts, 126, 393. See also
Standardized Service Contracts (principle)
APIs and, 129
auto-generation, 54, 152
in client-service applications, 128
content abstraction levels, 231-232
data models and, 134-137
defined, 126
denormalization and, 301-305
dependencies on, 165
design-time autonomy and, 301-305
designer autonomy and, 48
discoverability, 364-367
in distributed applications, 128
explained, 126-127
interpretability, 364-367
naming conventions, 133
non-technical contract documents,
Service Abstraction and, 237-238
normalization and, 301-305
runtime autonomy and, 301-305
Service Autonomy and, 301-305
service compositions and, 148
technical versus non-technical, 127
validation coupling and, 190-191
versions, 150
Web services architecture, 48
service coupling. See coupling
service custodians (role), 492
service description documents, explained, 126
service design
capability granularity and, 116
constraint granularity and, 117-118
data granularity and, 116
formal processes, design principles in, 106-107
granularity levels, 118
granularity types, 118
normalization and, 65
separation of concerns and, 70
service granularity and, 116
Service Reusability principle
design principles, relationship with, 278, 280-281
service granularity, 277-278
service models, 276-278
service-orientation principles and, 106-107
Service Discoverability (principle), 73, 243, 272, 276, 362-386. See also discovery
associated terminology, 486
in case study, 382-386
contribution to realizing ROI, 505
design principles, relationship with, 378-380
design risks
communication limitations, 381-382
post-implementation application, 381
discovery types, design-time and runtime discovery, 371-373
effect on other design principles, 378-380
explained, 362-364
implementation requirements, 370
interoperability and, 75
measuring
baseline measures checklist, 375-376
custom measures, 376
meta information types, 373
functional meta data, 374
quality of service meta data, 374
policy assertions and, 378
principle profile, 368, 370
Service Abstraction and, 241, 379
Service Composability and, 380, 436
service granularity and, 378
Service Loose Coupling and, 199
service modeling and, 106, 377-378, 525
Service Reusability and, 280, 380
service-oriented analysis processes and, 106
Standardized Service Contract and, 147-148, 379
support for service capability composition design process, 426
Web service region of influence, 370
service encapsulation, 235-237, 306
service enterprise models. See service inventory blueprints
service granularity, 486
cohesion and, 467
coupling and, 195-196
explained, 116
functional context and, 116
inheritance (OOAD) and, 473
Service Abstraction and, 238-239
Service Autonomy and, 311-312
Service Composability and, 427-428
Service Discoverability and, 378
Service Reusability, 277-278
Service Statelessness and, 346
standardization of service contracts, 142-144
service instances, 344-346
  Service Statelessness and, 344-346
service inventory. See also service inventory blueprints
  analysis process, 521
defined, 40
delivery processes, 520-521
evolutionary stages, 407, 409-410
modeling, 520-521
example of, 270
explained, 40
implementation, 42
as related to service compositions, 407
relationship with service-oriented computing elements, 41
service inventory blueprints, 53, 313, 320. See also service inventory architecture definition, 520
case study, 66
defining, 520
explained, 51-52
selecting processes, 521
Service Reusability, 269-270
service inventory models. See service inventory blueprints
service layers, 60, 82
service level agreement. See SLA
service logic autonomy, 306-308, 488
Service Loose Coupling (principle), 71, 164-209, 299. See also coupling
  associated terminology, 486
  association with Service Autonomy principle, 299
capability granularity and, 195-196
considerations when designing service-oriented classes, 473
constraint granularity and, 195-196
collection to realizing organizational agility, 506
  data granularity and, 195-196
effect on other design principles, 197-200
  interoperability and, 74
  performance, 202
  principle profile, 167, 169
Service Abstraction principle and, 114, 198, 241
Service Autonomy and, 178, 199, 315-316
Service Composability and, 199-200, 433
Service Discoverability principle and, 199
Service Reusability and, 199, 279
Standardized Service Contract principle and, 145-146, 173, 198
technology abstraction and, 221
Web services and, 50
service methods, explained, 115
service modeling, 60, 522-525. See also service-oriented analysis
  alternative terms for, 485
  business analysts and, 53
  business-centric, 45
classification, 485
coupling and, 196-197
entity services, 44
explained, 43-46, 52
non-business-centric, 46
orchestrated task services, 45
process, 523
Service Abstraction and, 239
Service Autonomy and, 105, 311-314, 525
service candidates, 52
Service Composability and, 428-430
Service Discoverability and, 106, 377-378, 525
Service Reusability and, 105, 276-278, 525
Service Statelessness and, 346-347
service-orientation principles and, 105-106
service-oriented design processes and, 526-527
standardization of service contracts, 144
task services, 44-45
technology architects and, 53
utility services, 46
wrapper service model, 306
Service Normalization design pattern, 272, 305, 465, 531
service contract autonomy and, 302-304
service operations, explained, 115
service policies, standardization of, 137-139
service profiles, 155
capability profiles, structure of, 481-482
case study, 155, 157
customizing, 482
design principles, relationship with, 278, 280-281
design risks, 281
explaining, 254
governance issues, 283-285
governing, 274-275
governance structure, 283-285
logic centralization design pattern
Enterprise design standard, 272
explained, 271
Web services and, 274
as enterprise design standard, 272
reliability, 286
security, 286
domain inventory design pattern
Effect on other design principles, 278-281
effect on other design principles, 279-281
explanation, 254
explanation, 254
interoperability and, 74
Logic Centralization design pattern
Contract Centralization and, 272-273
difficulty in achieving, 274-275
as enterprise design standard, 272
explained, 271
Web services and, 274
measuring, 262-263
in analysis/design phase, 265-266
commercial design approach, 262, 264-265
gold-plating, 267
in implementation phase, 267
principle profile, 259-261
change, 281
Service Reusability (principle), 62, 72,
agnostic services, 268-269
application level terminology, 487
in case study, 288-292
contribution to realizing organizational agility, 506
cultural issues, 281-283
design principles, relationship with, 278, 280-281
design risks, 281
agile delivery, 287
commercial design, 286-287
governance structure, 283-285
organizational culture, 281-283
reliability, 286
security, 286
domain inventory design pattern
Effect on other design principles, 278-281
explanation, 254
governance issues, 283-285
interoperability and, 74
Logic Centralization design pattern
Enterprise design standard, 272
explained, 271
Web services and, 274
measuring, 262-263
in analysis/design phase, 265-266
commercial design approach, 262, 264-265
gold-plating, 267
in implementation phase, 267
principle profile, 259-261
Index

reduced IT burden, 64
Service Abstraction and, 241, 279
Service Autonomy and, 280, 316
Service Composability and, 280, 435
service contracts and, 147
Service Discoverability and, 280, 380
service granularity, 277-278
service inventory blueprints, 269-270
Service Loose Coupling and, 199, 279
service modeling and, 105, 276-278, 525
Service Statelessness and, 280, 348
service-oriented analysis processes and, 105
Standardized Service Contract and, 147, 278
Web services and, 50
Service Statelessness (principle), 73, 326-359. See also state management in case study, 351-359
considerations when designing service-oriented classes, 473
contribution to realizing ROI, 505
design principles, relationship with, 347-349
design risks
architecture dependency, 349-350
runtime performance, 350
underestimating effort requirements, 350
effect on other design principles, 347-349
explained, 326
granularity and, 346
interoperability and, 74
measuring, 339
fully deferred state management, 342-343
internally deferred state management, 342
non-deferred state management, 340
partially deferred memory, 340-341
partially deferred state management, 341-342
messaging as deferral option, 343-344
principle profile, 331-332, 334
scalability, 261
Service Autonomy and, 316, 348
Service Composability and, 436
service instances and, 344-346
service models and, 346-347
Service Reusability and, 280, 348
state, types of, 335
active, 335
business data, 338
context data, 337-338
passive, 335
session data, 336-337
stateful, 336
stateless, 336
state deferral explained, 329
messaging as, 343-344
state delegation versus, 331
state delegation explained, 329
state deferral versus, 331
state management in client-server architectures, 328
databases and, 329, 331, 339-343
in distributed architectures, 329, 331
service symbol, explained, 13, 15-16

service-orientation

advantages of, 81-84
applications and, 82, 91-92
applications versus service compositions, 91-92
challenges introduced by, 85-88
counter-agile delivery and, 87
coupling types and, 193-195
defined, 39
design characteristics, importance of, 69
as design paradigm, 30, 70-71
design standards and, 86, 89
evolution of, 89
explained, 68-101
governance demands, 88
integration and, 84, 92-94
interoperability and, 74-75, 84
meta abstraction types in, 229-230
object-orientation compared, 97,
446-449
common goals, 449-452
design principles, 457-472
fundamental concepts, 453-457
origins of, 96-99
AOP (aspect-oriented programming), 99
BPM (business process management), 98
EAI (enterprise application integration), 98-99

object-orientation, 97
Web services, 98
problems solved by, 75-84
relationship with service-oriented computing elements, 40
reusability, level required, 90
service compositions, explained, 94-95
standardization and, 89
technology architectures and, 95-96
top-down delivery, 86-87

service-orientation principles. See also design principles
service modeling processes and, 105-106
service-oriented analysis processes and, 105-106
service-oriented design processes and, 106-107

service-oriented analysis, 60, 522-523.
See also service modeling
business analysts and, 53
design principles in, 105-106
explained, 52-53
process, 521
service-orientation principles and, 105-106
technology architects and, 53

service-oriented architecture. See SOA
Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services, 492
Service-Oriented Architecture: Concepts, Technology, and Design, 5, 100, 432, 518

service-oriented classes, designing, 472-474

service-oriented computing elements, 37-42
explained, 37-54
goals and benefits, 55-56
Index

business and technology domain alignment, 61
design principles, relationship with, 498-499
increased business and technology domain alignment, 60-61
increased federation, 58-59
increased intrinsic interoperability, 56-57
increased organizational agility, 63-64
increased ROI, 61-62
increased vendor diversification, 59
reduced IT burden, 64-65
as related to service-orientation principles, 104-105
relationships between, 56
vendor diversification, 59-60
governance, 88
implementation, 41-42
relationships among elements, 40-42
service compositions and, 39-40
service inventory and, 40
service inventory blueprints and, 51-52
service models and, 43-46
service-oriented analysis and, 52-53
service-oriented design and, 53-54
services and, 39
SOA and, 38, 56
terminology, 484-485
vision, 55
Web services and, 49-50
service-oriented design, 377, 521, 525, 527-528
contract first design, 53, 131, 173, 194
explained, 53-54
Service Abstraction
design principles, relationship with, 239-241
encapsulation, 235-237
non-technical contract documents, 237-238
service granularity and, 238-239
service models and, 239
Service Autonomy
design principles, relationship with, 314-317
service granularity and, 311-312
service models and, 311-314
Service Composability
composition autonomy and, 430
design principles, relationship with, 432-436
orchestration and, 430, 432
service granularity and, 427-428
service models and, 428-430
Service Contracts
data transformation, avoiding, 140-142
service granularity, 142-144
service models, 144
Service Discoverability
design principles, relationship with, 378-380
policy assertions and, 378
service granularity and, 378
service models and, 377-378
service models and, 526-527
Service Statelessness
design principles, relationship with, 347-349
granularity and, 346
messaging as deferral option, 343-344
service instances and, 344-346
service models and, 346-347
service-orientation principles and, 106-107
service-oriented solution logic
defined, 39
implementation, 42
relationship with service-oriented computing elements, 40
service-to-consumer coupling, 180
services
agnostic, 62, 82, 91
business-centric, 45
in case study, 154
as collections of capabilities, 69-70
communications quality, 365
as components, 176-177
as containers, 70
counter-agile delivery of, 87
defined, 39
dependencies between, 165
discoverability, 364-367
explained, 39, 68-69
as federated endpoints, 58
functional context, 70
implementation, 42, 47
interoperability, 84
interpretability, 364-367
as IT assets, 62
non-business-centric, 46
normalized, 65, 83
ownership, 88
real-world analogy, 68-70
relationship with service-oriented computing elements, 40
reusable versus agnostic, 268-269
reuse. See reuse; Service Reusability (principle)
ROI, 62
roles
service consumers, 48-49
service providers, 48-49
scalability, 326, 333, 340, 348
service candidates versus, 52
standardization of, 89
symbols for, 39
usage requirements, 318
Web services versus, 49
session data (state management), 336-337
shared autonomy, 305-306, 488
silo-based applications, 92
advantages of, 76-78
counter-federation and, 80
disadvantages of, 78-81
integration and, 81
redundancy, 78
Simple Object Access Protocol. See SOAP
single responsibility principle. See SRP
single-purpose programs, 255
SLA (service level agreement), 152-153, 237-238, 249, 382, 386, 483
SOA (service-oriented architecture), 5. See also service-oriented computing
explained, 38
goals and benefits, 498-499
governance, 88
relationship with service-oriented computing elements, 40
scalability, 326, 333, 340, 348
service-oriented computing versus, 56
vendor diversified, 60
vendor-agnostic, 60
Index

vision, 55
Web services and, 46-51
   architecture, 48-49
   standards, 47-48
SOA: Design Patterns, 4, 31-32, 111, 122,
   150, 474, 515, 530-531
The SOA Magazine Web site, 533
SOAP (Simple Object Access Protocol), 47
   attachments, 334, 344
   headers, 337-338, 344-346, 410
   processors, 334
software composition. See composition
   (OOAD)
specialization (OOAD), 461-462
SRP (OOAD), 466-468
standardization. See also standards
   functional expression, 147
   of service contracts
      data representation, 134-137,
         140-142, 155
      design principles, relationship
         with, 144-148
      functional service expression,
         133-134, 155
      service granularity, 142-144
      service models, 144
      service policies, 137-139
   of vocabularies, 484
Standardized Service Contract
   (principle), 71, 464
   agnostic service contracts and, 144
   capability granularity, 143
   case study, 154-161
   considerations when designing
      service-oriented classes, 473
   constraint granularity, 143
   coupling types, 169-173
   consumer-to-contract coupling,
      185-191, 214, 473, 486
   contract-to-functional coupling, 180
   contract-to-implementation coupling, 177-179
   contract-to-logic coupling, 174-175
   contract-to-technology coupling, 176-177
   logic-to-contract coupling, 173-174
data granularity, 143
design risks, 149
   development tool deficiencies, 151-152
   technology dependencies, 150
   versioning, 149-150
design standards and, 132
   effect on other design principles, 144-148
   functional meta data, 374
   origins of, 127-129
   interoperability and, 74
   naming conventions, 147
   non-agnostic service contracts
      and, 144
   non-technical service contracts
      and, 144
   principle profile, 130-132
   Service Abstraction and, 146, 240
   Service Autonomy and,
      301-305, 315
   Service Composability and,
      148, 432
   Service Discoverability and,
      147-148, 379
   Service Loose Coupling and,
      145-146, 173, 198
   service models and, 144
   Service Reusability and, 147, 278
standardization types
  data representation, 134-137, 140-142, 155
design principles, relationship with, 144-148
  functional service expression, 133-134, 155
  service granularity, 142-144
  service models, 144
  service policies, 137-139
  transformation and, 140-142
Web services and, 50
standards. See also design standards; standardization
  SOA, 5-6
  Web services standards, 47-48
  www.soaspecs.com Web site, 50
state, types of, 335
  active, 335
  business data, 338
  context data, 337-338
  passive, 335
  session data, 336-337
  stateful, 336
  stateless, 336
state data management. See state management
  state databases, 329, 331
state deferral
  explained, 329
  messaging as, 343-344
  state delegation versus, 331
state delegation
  explained, 329
  state deferral versus, 331
state management. See also Service Statelessness (principle)
in client-server architectures, 328
databases and, 329, 331, 339-343
in distributed architectures,
  329, 331
explained, 327-328
origins of, 328-331
performance and, 334
service compositions and, 340
SOAP attachments and, 334
state deferral and state delegation, 329, 331
state types, 335
  active, 335
  business data, 338
  context data, 337-338
  passive, 335
  session data, 336-337
  stateful, 336
  stateless, 336
stateful state (state management), 336
stateless state (state management), 336
statelessness. See Service Statelessness (principle)
static business process definition, explained, 397
Status (service profile field), 482
sub-classes (OOAD), 459, 461, 463
sub-controllers, explained, 398, 429
super-classes (OOAD), 459
symbols
  color in, 13
  conflict symbol, 13
  coupling, 165
  legend, 13
  service symbol, 13, 15-16, 39
T
tactical reusability, 487
  measuring, 265
targeted functional coupling, 180
targeted reusability, 487
  measuring, 266
Index

targeted reuse, example of, 289
task services, 340
coupling and, 197
example of, 44
explained, 44-45
functional coupling and, 180
Service Abstraction and, 239
Service Autonomy and, 313-314
service contracts, 144
in service inventory, 270
Service Statelessness and, 347
task-centric business services. See task services
technical communications specialists (role), 494
technical service contracts, explained in abstract, 127. See also service contracts
technology abstraction
Service Loose Coupling and, 221
Web services and, 221
technology and business alignment. See business and technology domain alignment in service-oriented computing
technology architects, role of, 53
technology architecture. See architecture
technology coupling, Contract Centralization design pattern, 189-190
technology dependencies of service contracts, 150
technology information abstraction, 219-221, 225, 486
technology services. See utility services
technology transformation, 142
terminology. See vocabularies
top-down processes, 86-87, 518-519

traditional solution delivery, explained, 76-81

transformation. See also data transformation
avoidance, 135-136, 140-142
design standards and, 135-136
standardization and, 140-142
Standardized Service Contract principle and, 135-136, 140-142
technology, 142

U

UDDI, 47, 367, 372
UML (unified modeling language), 447, 453
unidirectional coupling, 165
Universal Description, Discovery, and Integration. See UDDI
uses-a relationships (OOAD), 469, 471, 474
utility services
coupling and, 197
design processes, 526
explained, 46
Service Abstraction and, 239
Service Autonomy and, 313
service contracts, 144
Service Statelessness and, 347

V

Validation Abstraction design pattern, 531
validation coupling, 190-191
performance and, 202
validation logic
constraint granularity and, 117-118
policies, 137
vendor diversification in service-oriented computing, 59-60
Version (service profile field), 482
versioning, 260, 438
  service contracts and, 149-150
vocabulary, 147
  for design principle application levels, 487-488
  for design principles, 486-487
  for policy assertions, 137-138
  service contracts, 133
  service models, alternative terms for, 485
  service-oriented computing terminology, 484-485
  standardization of, 484

Web Service Contract Design for SOA, 5, 150, 153

Web service regions of influence
  composition members, 395
designated controllers, 396
  functional abstraction, 225
  programmatic logic abstraction, 226
  quality of service abstraction, 226
  service autonomy, 297
  service contracts, 131-132
  service discoverability, 370
  service loose coupling, 169
  service reusability, 260-261
  service statelessness, 334
  technology information abstraction, 225

Web services, 46-51
  architecture, 48-49
  auto-generation of contracts, 54, 152, 175
  avoiding technology dependencies, 150
  consumer-to-contract coupling and, 186

  Contract Centralization design pattern and, 190, 274
  contracts, 134-137
  coupling and, 166
design processes, 527
  federation via, 59
  first-generation platform, 47
  implementation coupling and, 166
  as implementation medium, 114
  as industry standards, 34
  as influence of service-orientation, 98, 448
  interface element, 456
  Logic Centralization and, 274
  logic-to-contract coupling and, 201
  meta abstraction types and, 225-226, 229-230
  origins of reuse, 258
  origins of Standardized Service Contract principle, 129
  policies. See policies; WS-Policy
  portType element, 456
  reuse and, 258
  roles
    service consumers, 48-49
    service providers, 48-49
  Schema Centralization design pattern and, 135-137
  schema custodians (role), 492
  second-generation platform, 47
  service compositions and, 401, 405-406
  service contracts, 127
  service description documents, 127
  service-oriented computing, 49-50
  services versus, 49
  standardization, 134-137
  standards, 47-48
technology abstraction and, 221
Index

technology-to-contract coupling and, 177
validation coupling and, 190-191
Web Services Business Process Execution Language. See WS-BPEL
Web Services Description Language. See WSDL
Web services tutorials Web site, 50, 534
Web sites
  www.soabooks.com, 16, 531, 533
  www.soaglossary.com, 16, 533
  www.soamag.com, 17, 533
  www.soaposters.com, 16, 534
  www.soaspecs.com, 16, 338, 460, 493, 533
  www.thomaserl.com, 17
  www/xmlenterprise.com, 534
wrapper services, 306, 316
  Service Autonomy design risks, 318
WS-* extensions, 47, 395
WS-Addressing, 337-338, 344-346, 373
WS-AtomicTransaction, 338
WS-BPEL, 197, 239, 431-432, 527
WS-Coordination, 338
WS-I Basic Profile, 47, 150-151
WS-MetadataExchange, 372-373
WS-Policy, 48, 129, 131, 483, 493
WS-Policy definitions, 127, 137-139, 146, 151, 153, 274
  contract-to-logic coupling, 179
  editors, 152
  structural standards, 139
  wsp:optional attribute, 139
WS-ReliableMessaging, 137
WS-ResourceTransfer (WS-RT), 338
WS-SecurityPolicy, 137
WSDL (Web Services Description Language), 47, 129, 131, 146, 174, 274
WSDL definitions, 127, 175
  auto-generation, 175
  standardization, 136
XML schemas and, 136
wsp: ignorable attribute, 143, 238, 378
wsp:optional attribute, 139, 143, 238, 378
X—Z
XML, 194
  as industry standards, 34
  parsers, 334
XML Schema Definition Language, 47, 129, 131
XML schemas, 127, 137, 146, 174-175, 274, 455
  case study, 157
  constraint granularity example, 117
  entity schemas, 136
  schema custodians (role), 492
  standardization, 136
  validation coupling and, 190-191
  WSDL definitions and, 136
XML tutorials Web site, 534
XSD. See XML Schema Definition Language; XML schemas
XSLT, 140