



Index

- Abbreviations in class names, 137
- abs operation, 508
- Abstract classes, 391
- Abstraction
 - in class inheritance, 210–211
 - dependencies in, 198–200
 - levels of, 211
- Accept action events
 - in activity diagrams, 314–317
 - time, 296–297
- Access
 - for collections, 515
 - scope for, 147–148
- «access» dependency, 200, 229–230
- Acronyms in class names, 137
- Action nodes in activity diagrams, 286
 - accept time event, 296–297
 - call, 294–296, 536
 - execution of, 293–294
 - with tokens, 288–289
- Actions
 - for states, 444–445
 - for transitions, 446
 - in use case modeling, 111
- Activation
 - in lifelines, 246
 - in sequence diagrams, 253
- Active classes in concurrency, 420–422
- Activities
 - in activity diagrams. *See* Activity diagrams
 - for states, 443–445
 - in UP and RUP, 33
- Activity diagrams, 283–284
 - action nodes in
 - accept time event, 296–297
 - call, 294–296
 - execution of, 293–294
 - with tokens, 288–289
 - activities in, 286–288
 - partitions, 290–293
 - semantics, 288–289
 - advanced, 309–310
 - central buffer nodes in, 322–323
 - connectors in, 311
 - control nodes in, 286, 297
 - decision and merge, 298–299
 - fork and join, 299–301
 - initial and final, 298
 - events in, 314–317
 - exception handling in, 312–313
 - expansion nodes in, 313–314
 - features of, 284–285
 - interaction overview, 323–325
 - interruptible activity regions in, 311–312
 - multicast and multireceive in, 320–321
 - object flow features in, 318–320
 - object nodes in, 286, 301–302
 - activity parameters in, 304–306
 - buffers for, 302–303
 - pins in, 305–306
 - state representation in, 303
 - OCL in, 536–537
 - parameter sets in, 321–322
 - signals in, 314–317
 - streaming in, 317–318
 - summary, 307–308, 325–327
 - in unified process, 285–286
- Activity parameters in object nodes, 304–306
- Actor classifiers, 19
- Actors in use case modeling, 69–71
 - features of, 71–72
 - generalization of, 97–99

570 Index

- Actors in use case modeling, *continued*
 - identifying, 72–73
 - in specification, 80
 - time as, 73
- Adornments, 17–18, 136
- after keyword, 453
- Aggregation, 134
 - vs. inheritance, 351–352
 - in relationships
 - features, 363–364
 - semantics, 364–367
- Algorithms, plug-in, 405
- allInstances operation, 505–506
- alt operator, 257
 - branching with, 258–261
 - with continuations, 281
- Alternative flows, 85–88
 - finding, 89
 - number of, 89–90
 - in use case generalization, 99
- Alternative sets, 321
- Analysis classes, 155–158
 - in analysis packages, 224
 - diagrams for, 243–244
 - features of, 158–159
 - finding
 - archetype patterns for, 170–171
 - by CRC analysis, 165–167
 - by noun/verb analysis, 163–165
 - by RUP stereotypes, 167–169
 - first-cut analysis models, 171
 - good, 160–161
 - parts of, 159–160
 - rules of thumb, 162–163
 - summary, 172–173
- Analysis models
 - and design models, 334
 - in use case realization design, 416
- Analysis packages. *See* Packages
- Analysis relationships. *See* Relationships
- Analysis workflow, 119
 - activity diagrams for, 286
 - artifacts in, 121
 - detail in, 122
 - features of, 120
 - models for, 122–123
 - summary, 124
- and operator, 507
- AndromDA tool, 9
- any operator, 519
- append operator, 518
- Archetype patterns, 170–171
- Architecturally significant components, 482
- Architecture, 23–24
 - analysis, 231–235
 - in design workflow, 338–339
 - implementation, 482–483
 - and layering patterns, 406–407
- ArcStyler tool, 9
- Arrow operator (->) for collection operations, 513
- «artifact» stereotype, 487–488
- Artifacts, 33
 - in analysis workflow, 121
 - of components, 400, 486
 - in deployments, 486–489
 - in design workflow, 333–335
 - in implementation workflow, 477–479
 - trace relationships, 335
- asBag operation, 514
- asOrderedSet operation, 514
- asSequence operation, 514
- assert operator, 258
- asSet operation, 514
- Associations
 - in aggregation, 366
 - attribute links to, 190–191
 - bidirectional, 376–377
 - classes for, 191–193, 377–378
 - of components, 400

- features of, 180–181
 - inherited, 542–544
 - for interfaces, 404
 - multiplicity in, 182–187
 - navigability of, 187–189
 - in OCL navigation, 523–526, 540–543
 - qualified
 - features of, 193–194
 - navigation through, 541
 - reflexive, 185–186
 - in relationships, 369
 - many-to-many, 375–376
 - many-to-one, 370–371
 - one-to-many, 371
 - one-to-one, 369–370
 - syntax of, 181–182
- Asterisks (*)
- in communication diagrams, 265–266
- Asymmetry
- in aggregation, 365–366
 - of composition, 367
- Asynchronous communication
- for submachines, 467
 - in use case realizations, 246–247
- at operation, 516
- Atomic operations
- in concurrency, 424
 - in design classes, 348
- attach operation, 546
- Attributes
- analysis class, 159
 - association links to, 190–191
 - component, 400
 - for composition, 368
 - constraints on, 256
 - design class, 344–346
 - in interfaces, 390
 - notation for, 142
 - object, 132
 - of requirements, 59–60
 - scope, 147–148
 - state, 443
 - visibility, 138–139
- Automatic transitions, 446
- Axioms, 34–35
- Backplanes, semantic, 16
- Bags, 374
- Base use cases, 103
- Baselines, 37
- Behavioral state machines, 440
- Behaviors
- call action nodes for, 294–296
 - of object nodes, 303
 - of objects, 127–129
 - reusing, 465
- Benefit attribute, 60
- Bidirectional associations, 376–377
- Bidirectional links, 178
- Bidirectional relationships, 234–235
- «bind» stereotype, 355–356
- Binding, 355
- Black boxes
- components as, 399–400
 - subsystems as, 426–427
- body: expression, 503, 530–531
- Booch, Grady, 31
- Booch method, 5–6
- Boolean type
- for change events, 452
 - features of, 507–508
 - iteration operations, 519–520
 - semantics of, 140
- Boundaries
- packages for, 224
 - in use case modeling, 71
- «boundary» stereotype, 167–168
- Brainstorming
- in CRC analysis, 166
 - for requirements, 64

572 Index

- Branching
 - in communication diagrams, 267–268
 - in interaction overview diagrams, 324
 - in main flow, 82–85
 - with opt and alt, 258–261
 - transitions, 448
- break operator
 - iteration with, 261–263
 - semantics of, 257
- Brief descriptions, 80
- Buffers
 - in activity diagrams, 322–323
 - for object nodes, 302–303
- «buildComponent» stereotype, 402
- Building blocks, 11–15
- Business models
 - activity diagrams for, 285–286
 - in use case modeling, 70
- C++ language
 - abstract classes in, 391
 - constructors in, 148, 247–248
 - destructors in, 150, 247–248
 - inheritance in, 351
 - interface names in, 392
 - links in, 177
 - overriding in, 209
 - primitive types in, 368
 - template support for, 356
- C# language
 - constructors in, 148, 247–248
 - destructors in, 150, 247–248
 - inheritance in, 351–352
 - template support for, 356
- Call action nodes, 294–296, 536
- «call» dependencies, 197
- Call events, 449
- CBD (component-based development), 399
- «centralBuffer» nodes, 322–323
- «centralBuffer» stereotype, 303
- Change events, 452–453
- Child packages, 231
- Choice pseudo-states for transition
 - connections, 448
- CIMs (computer-independent models), 8
- Class classifiers, 19
- Class generalization, 207
- Class scope, 147
- Class style notation for interfaces, 391–392
- Classes, 125–126, 132–134
 - abstract, 391
 - activity partitions for, 291
 - analysis. *See* Analysis classes
 - association, 191–193, 377–378
 - in CRC analysis, 165–167
 - dependencies between, 195–196
 - design. *See* Design classes
 - designing, 342–344
 - inheritance in, 208–211, 350–354
 - instantiation of, 135
 - loops for, 263
 - moving between packages, 233
 - notation for. *See* Notation
 - in object names, 132
 - and objects, 134–135
 - operations for, 142–146
 - partitioning, 219
 - powertypes for, 218–220
 - relationships with. *See* Relationships
 - state machines for, 440–441
 - stereotype syntax for, 146
 - structured, 378–382
 - summary, 151–154
 - templates for, 354–356
- Classification, 133–134
- Classifiers
 - divisions, 18
 - structured, 378–379
 - in use case realizations, 244
- Cohesion
 - for analysis packages, 233
 - in design classes, 349

- Collaboration
 - messages for, 130
 - in structured classifiers, 379
- Collaborators in CRC analysis, 165–167
- collect operator
 - for collections, 525
 - for iterators, 519
- Collections, 141
 - features of, 511–513
 - loops for, 263
 - maps, 374
 - operations for, 371–372, 513
 - access, 515
 - comparison, 514
 - conversion, 514
 - query, 514–515
 - selection, 516–518
 - working with, 371–374
- collectNested operator
 - for collections, 525
 - for iterators, 519
- Colons (:) in names
 - action, 292
 - object, 132
 - package, 226
 - powertype, 219
- Color in UML models, 20
- Combined fragments, 256–257
 - branching in, 258–261
 - iterations in, 261–263
- Commas (,) for actions, 292
- Comments, 504–505
- Common mechanisms
 - adornments, 17–18
 - divisions, 18–19
 - extensibility, 19–22
 - specifications, 15–17
- Communication. *See* Messages
- Communication diagrams, 264–265
 - branching in, 267–268
 - concurrency in, 425–426
 - iteration in, 265–266
- Comparison operations
 - for collections, 514
 - in OCL, 506
- {complete} constraint, 217
- Complete models, 17
- Completeness in design classes, 347–348
- Complex deviations in specifications, 81
- Complex nesting of packages, 227
- Complexity with interfaces, 408
- Component-based development (CBD), 399
- Component-based modeling, 170–171
- Component classifiers, 19
- Components, 389–390
 - activity partitions for, 291
 - architecturally significant, 482
 - from artifacts, 486
 - features of, 399–402
 - in interfaces, 399
 - stereotypes, 402
 - subsystems, 403
 - summary, 410–411
- Composite icons, 459
- Composite states
 - features of, 458–460
 - orthogonal, 462–465
 - simple, 460–462
- Composite structured diagrams, 381
- Composition, 134
 - attributes for, 368
 - in relationships, 363–364, 367–368
 - with structured classes, 378–382
- Computer-independent models (CIMs), 8
- concat operator, 509
- Conceptual entities, classes for, 170
- Concrete operations
 - vs. abstract, 210
 - overriding, 214
- Concurrency, 419
 - active classes in, 420–422
 - in communication diagrams, 425–426
 - fork and join control nodes for, 299–301

574 Index

- Concurrency, *continued*
 - in interaction overview diagrams, 324
 - in sequence diagrams, 422–425
- Conditional extensions, 109
- Conditions in communication diagrams, 267
- Configuration management, packages for, 224
- Connecting transitions for state machine, 447
- Connectors and links, 177–178
 - in activity diagrams, 311
 - in communication diagrams, 265
 - of components, 400–401
 - in object diagrams, 178–179
 - paths for, 179–180
 - in structured classifiers, 378–379, 381
- consider operator, 258
- Consistent models, 17
- Constraint languages, 499
- Constraints, 20
 - in association classes, 378
 - on expansion nodes, 313–314
 - generalization, 217
 - in interfaces, 390
 - multiplicity, 182–187
 - in OCL, 503
 - in sequence diagrams, 254–256
 - in structured classifiers, 379–380
 - in timing diagrams, 427
- Construction phase, 37–39, 41–43, 476
- Constructor operations, 135, 148–150, 247–248
- Context classifiers, 244
- Context-free questions in requirements interviews, 63
- Context in OCL
 - expression, 501–502
 - package, 500–501
- Contextual instances, navigation within, 522–523
- Continuations in use case realizations, 279–281
- Continuous streaming in activity diagrams, 317–318
- Contracts
 - abstract classes as, 210
 - interfaces as, 391
 - in polymorphism, 212
- Control flows in activity diagrams, 286
- Control nodes in activity diagrams, 286, 297
 - decision and merge, 298–299
 - fork and join, 299–301
 - initial and final, 298
- «control» stereotype, 168–169
- Conversion operations
 - for collections, 514
 - in OCL, 506
- Could have requirements, 59
- count operator, 515
- Coupling
 - in analysis classes, 161
 - in analysis models, 123
 - in analysis packages, 233
 - in architectural analysis, 232
 - in design classes, 350
 - inheritance in, 351
- CRC analysis, finding classes by, 165–167
- «create» stereotype, 248
- Creation messages in use case realizations, 247–248
- critical operator, 257
- Curved paths for links, 180
- Cyclic package dependencies, 234–235
- Data-hiding, 127–130
- Decision control nodes, 298–299
- «decisionInput» stereotype, 298
- Declarative languages, 498
- Decomposition, functional, 112–113
- Decoupling
 - components, 400–401
 - layers, 406
- Deep history in state machines, 469–470
- Deep inheritance trees, 162

- def: expression, 503, 531–533
- Default values in parameters, 145
- Define operation, 503
- Deleting
 - filters for, 61–62
 - packages, 233
- Dependencies, 195–196
 - in abstraction, 198–200
 - «derive», 199
 - «refine», 199
 - «substitute», 198
 - «trace», 198, 229
 - in aggregation, 365
 - in bidirectional associations, 376–377
 - of components, 400–401
 - in composition, 367
 - interfaces for, 408
 - between layers, 406
 - of packages, 228–230, 234–235
 - permission, 200
 - «access», 200, 229–230
 - «import», 200, 229
 - «permit», 200
 - usage, 196–198
 - «call», 197
 - «instantiate», 134–135, 198
 - «parameter», 197–198
 - «send», 198
 - «use», 196–197, 228–229
- Deployment, 481
 - architectural implementation, 482–483
 - artifacts in, 486–489
 - diagrams for, 483–484
 - example, 491
 - nodes in, 484–485
 - summary, 491–492
 - in use case realization design, 416
 - views, 23
 - «deployment spec» stereotype, 488
- derive: expression, 503, 533–534
- «derive» stereotype, 199
- Derived value operator, 503
- Descriptions in specifications, 80
- Descriptor form of deployment diagrams, 483
- Design and design workflow, 331
 - architectural design in, 338–339
 - artifacts in, 333–335
 - for classes, 342–344
 - to contracts, 391
 - detail in, 337–338
 - features of, 332–333
 - with interfaces, 404–407
 - in iteration workflow, 36
 - multiple models in, 336–337
 - relationships in, 335, 363
 - of subsystems, 389
 - summary, 339–340
 - in use case realization design, 416
- Design classes, 341–342
 - anatomy of, 345–347
 - features of, 344–345
 - inheritance in, 350–354
 - nested, 357
 - summary, 358–360
 - templates for, 354–356
 - well-formed, 347–349
- «destroy» stereotype, 248
- Destruction messages, 247–248
- Destructor operation, 148–150, 247–248
- detach operation, 546
- Detail
 - in analysis workflow, 122
 - in design workflow, 337–338
 - in implementation workflow, 478–479
 - in requirements workflow, 53–54
 - in use cases, 77–78
- Deviations in specifications, 81
- Device boundary classes, 167–168
- «device» stereotype, 484
- Diagrams, 12–15
 - activity. *See* Activity diagrams
 - analysis class, 243–244
 - class, 136–137

576 Index

- Diagrams, *continued*
communication, 264–265
 branching in, 267–268
 concurrency in, 425–426
 iteration in, 265–266
component, 400
composite structure, 381
deployment, 483–484
interaction
 in design, 417–419
 OCL in, 535–536
 in use case realizations, 243–244,
 248–249
navigability on, 187–189
sequence
 concurrency in, 422–425
 interaction in, 274–276
 in use case realizations, 249–256
state machine, 442–443
timing, 427–430
use case, 75
- Dictionaries
 maps, 374
 project glossaries, 75–77
«directory» stereotype, 489
- Disciplines, 33
- {disjoint} constraint, 217
- Distortion filters, 61–62
- div operation, 508
- Divisions, 18–19
- do keyword, 445
- «document» stereotype, 488
- Documenting sequence diagrams, 253–254
- DOORS tool, 56–57, 90
- Dot operator
 in navigation, 524–525
 for tuples, 510
- Duration
 accept time event action nodes for, 297
 in sequence diagrams, 256
- Dynamic connections, links as, 178
- Eclipse Modeling Framework, 9
- Edges
 in activity diagrams, 286
 with tokens, 289
- Effort attribute, 60
- «EJB» stereotypes, 489
- EJBs (Enterprise JavaBeans), 400
- Elaboration phase, 37–41, 476
- Elicitation for requirements, 61–62
- Elided models, 17
- Embedded systems, concurrency in, 420
- Encapsulated namespaces, packages for, 224
- Encapsulation, 127–130
- Enterprise JavaBeans (EJBs), 400
- «entity» stereotype, 169, 402
- Entry events, 445
- Equal signs (=)
 for Boolean type, 507
 for collection comparisons, 516
 for OCL comparisons, 506
 for strings, 509–510
- Equivalent objects, 528
- Events
 accept action
 in activity diagrams, 314–317
 time, 296–297
 in state machines, 439–440, 448–449
 call, 449
 change, 452–453
 signal, 450–451
 time, 453
 transition, 446
- Exception handling in activity diagrams,
 312–313
- excludes operator, 515
- excludesAll operator, 515
- excluding operator, 517
- «executable» stereotype, 488
- «execution environment» stereotype, 484
- exists operator, 519
- Exit events, 445

- Expansion nodes in activity diagrams, 313–314
- Explicit binding, 355
- Expressions in OCL
 - bodies, 504
 - collections. *See* Collections
 - comments, keywords, and precedence rules, 504–505
 - infix operators in, 511
 - iteration operations, 518–522
 - primitive types in, 506–509
 - tuples in, 509–510
 - type system, 505–506
- body:, 530–531
- context, 501–502
- def:, 531–533
- derive:, 533–534
- init:, 531
- inv:, 526–529
- let, 533
- OclMessage, 544–546
- pre:, post:, and @pre, 529–530
- syntax, 498–500
- types, 502–503
- «extend» stereotype
 - conditional, 109
 - features of, 105–107
 - multiple insertion segments in, 108–109
- Extensibility mechanisms, 19–22
- Extension points, 99
- «external» stereotype, 292

- Facade patterns, 405–406
- Feature lists, 70
- FIFO (first-in, first-out) buffers, 303
- «file» stereotype, 488
- Filters, 61–62
- Final control nodes, 298
- Find actors
 - in requirements workflow, 54
 - in use case modeling, 69–77

- Finding
 - alternative flows, 89
 - analysis classes
 - archetype patterns for, 170–171
 - by CRC analysis, 165–167
 - by noun/verb analysis, 163–165
 - by RUP stereotypes, 167–169
 - analysis packages, 232–234
 - interfaces, 404
 - requirements, 61–63
- First-cut analysis models, 171
- First-in, first-out (FIFO) buffers, 303
- first operation, 516
- Flags for submachine communication, 467
- flatten operator, 514
- Flexibility
 - of components, 400–401
 - of interfaces, 396–397, 408
- Floating-point numbers
 - semantics of, 140
 - working with, 508–509
- floor operation, 509
- Flows
 - in activity diagrams, 286, 318–320
 - alternative, 85–90
 - branching in, 82
 - if keyword, 83
 - for keyword, 84
 - while keyword, 85
 - in interaction, 246
 - in specifications, 81–85
 - tokens for, 288–289
- Focus of control
 - in lifelines, 246
 - in sequence diagrams, 253
- For loops
 - in main flow, 84
 - in sequence diagrams, 263
- forAll operator, 520
- forEach operator, 263
- Fork control nodes, 299–301

578 Index

- Found messages in use case realizations, 248
- Fragile base class problem, 351
- Fragments in use case realizations, 256–257
 - branching in, 258–261
 - iterations in, 261–263
- Frames, 12
- «framework» stereotype, 226
- Functional decomposition, 112–113
- Functional requirements, 57–58
- Functoids, 162
- Fusion method, 6

- Garbage collection, 248
- Gates in interaction occurrences, 277–278
- Generalization, 206–207
 - class, 207
 - packages, 231
 - powertypes, 218–220
 - sets, 216–218
 - in use case modeling
 - actor, 97–99
 - use case, 99–102
- Generalization filters, 61–62
- Generic form interaction diagrams, 245
- Glossaries, 75–77
- Greater than signs (>)
 - for Boolean type, 507
 - for collection comparisons, 516
 - for OCL comparisons, 506
 - for strings, 509
- Groups
 - for analysis packages, 232–234
 - port, 398–399
- Guard conditions
 - in activity diagrams, 536
 - for decision nodes, 298–299
 - for fragments, 256–257
 - in interaction diagrams, 535
 - in state machines, 537
 - for transitions, 446
- HashMap class, 374
- hasReturned operator, 544
- Helper operations, 531–533
- Hierarchies in associations, 186–187
- High cohesion, 349
- History, state machine, 468–470
- Homonyms, 76

- Identical objects, 528
- Identifying
 - actors, 72–73
 - use cases, 74–75
- Identity property, 127
- IDs, use case, 80
- If keyword
 - Boolean expressions for, 508
 - in main flow, 83
- ignore operator, 258
- Immutable collections, 512
- Immutable strings, 508
- «implementation» stereotype, 402
- Implementation, 19
 - architectural, 482–483
 - vs. specifications, 389–391
 - views, 23
 - workflow, 475–476
 - artifacts in, 477–479
 - details in, 478–479
 - features of, 476–477
 - summary, 480
- implies operator, 507
- «import» dependencies, 200, 229
- in parameter, 144
- Inception phase, 37–40
- «include» stereotype, 102–105
- includes operator, 515
- includesAll operator, 515
- including operator, 517
- {incomplete} constraint, 217
- Incomplete models, 17
- Inconsistent models, 17
- Incremental processes, 35–37

- indexOf operation, 516
- Infix operators, 511
- Inheritance, 205–206
 - abstract operations in, 210–211
 - vs. aggregation, 351–352
 - of associations, 542–544
 - class, 208–211
 - in design classes, 350–354
 - vs. interface realization, 353–354, 394–398
 - multiple, 211, 352–353
 - overriding, 99–100, 208–209
 - summary, 221–222
 - in use case generalization, 99–100
- Inheritance trees, 162
- init: expression, 503, 531
- Initial control nodes, 298
- Initial values
 - notation for, 141–142
 - operation for, 503, 531
- inout parameter, 144
- Input effects in activity diagrams, 319–320
- Inputs for interactions, 277–278
- insertAt operator, 518
- Insertion segments, multiple, 108–109
- Instance form of diagrams
 - deployment, 483
 - interaction, 245
- Instance scope, 147
- Instances
 - artifact, 486
 - of association classes, 193
 - contextual, 501, 522–523
 - features of, 18
 - navigation within, 522–523
 - objects as, 133
- «instantiate» stereotype, 134–135, 198
- Instantiation
 - class, 135
 - template, 355–356
 - unified process, 34
- Integer type
 - semantics of, 140
 - working with, 508–509
- Interaction diagrams
 - in design, 417–419
 - OCL in, 535–536
 - overview, 323–325
 - in use case realizations, 243–244, 248–249
- Interactions
 - interaction occurrences, 274–276
 - continuations with, 279–281
 - gates in, 277–278
 - parameters in, 276–277
 - subsystems, 426–427
 - in use cases realizations, 244
- Interface boundary classes, 167–168
- Interface classifiers, 19
- Interface realization vs. inheritance, 353–354, 394–398
- Interfaces, 19, 389–390
 - advantages and disadvantages, 408
 - for component-based development, 399
 - component stereotypes, 402
 - designing with, 404–407
 - features of, 389–391
 - finding, 404
 - ports for, 398–399
 - provided and required, 391–393
 - summary, 408–411
- Internal structure of components, 400–401
- Internal visibility, 139
- Interruptible activity regions, 311–312
- intersection operation, 516
- Interval specifications for sequences, 512
- Interviews for requirements, 63
- inv: expression, 503, 526–529
- invariant operator, 503, 526–529
- "is a" relationships, 350
- "is kind of" principle, 352–353
- isEmpty operator, 515
- isOperationCall operator, 544
- isSignalSent operator, 544

580 Index

- isUnique operator, 519
- iterate operator, 521–522
- Iteration
 - in communication diagrams, 265–266
 - in interaction overview diagrams, 324
 - with loop and break, 261–263
 - operations for, 518–522
 - in unified process, 35–37
- Iterative expansion nodes, 314
- iUML tool, 9

- J2EE server, 490–491
- Jacobson, Ivar, 29–31
- JAR (Java ARchive) files, 400
 - «JAR» stereotype, 489
- Java language
 - collections in, 372
 - constructors in, 148, 247–248
 - destructors in, 150, 247–248
 - inheritance in, 351–352
 - interfaces in, 391
 - links in, 177
 - nested classes in, 357
 - overriding in, 209
 - primitive types in, 368
 - profiles for, 489
 - standard libraries in, 393, 407
 - template support for, 356
 - «JavaClassFile» stereotype, 489
 - «JavaSourceFile» stereotype, 489
- JMechanic for Java tool, 348–349
- Join control nodes, 299–301
- Junction pseudo-states for transition connections, 447

- Keys for maps, 374
- Keywords
 - in expressions, 504
 - in OCL, 504–505

- Languages
 - for analysis models, 123
 - constraint, 499
 - declarative, 498
 - filters in, 61–62
- Last-in, first-out (LIFO) buffers, 303
- last operation, 516
- Layers in architecture
 - arranging, 231–232
 - patterns in, 406–407
- Less than signs (<)
 - for Boolean type, 507
 - for collection comparisons, 516
 - for OCL comparisons, 506
 - for strings, 509
- let expression, 503, 533
- Levels of abstraction, 211
- Libraries
 - in Java, 393, 407
 - wxPython, 433–434
 - «library» stereotype, 488
- Lifelines
 - in communication diagrams, 265
 - constraints on, 256
 - in continuations, 280–281
 - with interaction occurrences, 276
 - messages for, 246
 - in sequence diagrams, 249–252
 - in timing diagrams, 427–429
 - in use case realizations, 244–245
- LIFO (last-in, first-out) buffers, 303
- Links and connectors, 177–178
 - in activity diagrams, 311
 - in communication diagrams, 265
 - of components, 400–401
 - in object diagrams, 178–179
 - paths for, 179–180
 - in structured classifiers, 378, 381
- Listening in requirements interviews, 63
- Logical groupings, packages for, 224
- Logical views, 23–24
- Lollipop style notation, 391–392
- Loop operations, 257
 - iteration with, 261–263
 - in main flow, 83–85
- Lost messages in use case realizations, 248

- Low coupling in design classes, 350
- lowerCamelCase naming convention, 132, 138, 143
- Main flow
 - alternative, 85–90
 - branching in, 82
 - if keyword, 83
 - for keyword, 84
 - while keyword, 85
 - in specifications, 81–85
- «manifest» relationship, 478
- Many-to-many associations, 375–376
- Many-to-one associations, 370–371
- Maps, 374
- max operation, 508
- MDA (Model Driven Architecture), 7–9
- Merge control nodes, 298–299
- «merge» stereotype, 230
- Messages
 - in diagrams
 - activity, 535–536
 - communication, 265
 - sequence, 249–252
 - timing, 429
 - with interaction occurrences, 276
 - for interfaces, 404
 - for objects, 130–131
 - in polymorphism, 213–214
 - for submachines, 467
 - in use case realizations, 246–248
- Metaclasses, 218
- Methods, 128
- min operation, 508
- Minus signs (-)
 - for comments, 504
 - for visibility, 138
- mod operation, 508
- Model Driven Architecture (MDA), 7–9
- «modelLibrary» stereotype, 226
- Models
 - for alternative flows, 85–90
 - for analysis workflow, 122–123
 - for collection classes, 372–373
 - in design workflow, 336–337
 - for requirements, 56
 - use case. *See* Use case modeling
- MoSCoW criteria, 59–60
- Moving classes between packages, 233
- Multicast in activity diagrams, 320–321
- Multiple associations, navigation across, 525–526
- Multiple inheritance, 211, 352–353
- Multiple insertion segments, 108–109
- Multiple models in design workflow, 336–337
- Multiplicity
 - in associations, 182–185
 - hierarchies and networks, 186–187
 - reflexive, 185–186
 - notation for, 140–141
 - of ports, 399
 - in relationships, 369
 - in structured classifiers, 378
- Multireceive in activity diagrams, 320–321
- Multithreading, 422
- Must have requirements, 59
- Named sets of public features, interfaces for, 389
- Names
 - analysis classes, 159–161
 - associations, 181–182
 - classes, 137
 - composite structure diagrams, 381
 - in constructors, 148
 - interfaces, 392
 - lifelines, 244
 - objects, 131–132
 - operations, 142–143
 - package elements, 500–501
 - packages, 224
 - ports, 398
 - powertypes, 219
 - qualified, 226–227
 - role, 181–182, 378
 - tuple parts, 509
 - use case, 79–80

582 Index

- Namespaces, packages for, 224, 226–227
- Navigability
 - of associations, 187–189
 - in relationships, 369
- Navigation in OCL, 523
 - to and from association classes, 540–541
 - across associations, 523–526
 - within contextual instances, 522–523
 - through qualified associations, 541
- neg operator, 258
- Nested items
 - classes, 357
 - collections, 513
 - components, 400–401
 - nodes, 484
 - packages, 227–228
 - states, 459
- Networks in associations, 186–187
- Node classifiers, 19
- Nodes, 286, 536–537
 - action
 - accept time event, 296–297
 - call, 294–296
 - execution of, 293–294
 - with tokens, 288–289
 - central buffer, 322–323
 - control, 286, 297
 - decision and merge, 298–299
 - fork and join, 299–301
 - initial and final, 298
 - in deployment, 484–485
 - expansion, 313–314
 - object, 286, 301–302
 - activity parameters in, 304–306
 - buffers for, 302–303
 - pins in, 305–306
 - state representation in, 303
- Non-functional requirements, 57–58
- {nonunique} property, 373–374
- not operator, 507–508
- Notation
 - for classes, 136–137
 - advanced attribute syntax, 142
 - attributes, 138–142
 - initial values, 141–142
 - multiplicity, 140–141
 - name convention for, 137
 - operations, 142–146
 - stereotypes, 146
 - type, 140
 - for objects, 131–132
- notEmpty operation, 515
- notify operation, 546
- Noun/verb analysis, finding classes by, 163–165
- Null values, 141
- Number signs (#) for visibility, 138
- Numbering in communication diagrams, 265
- Object Constraint Language. *See* OCL (Object Constraint Language)
- Object flows, 286, 301, 318–320
- Object Modeling Technique (OMT), 5–6
- Object nodes in activity diagrams, 286, 301–302
 - activity parameters in, 304–306
 - buffers for, 302–303
 - pins in, 305–306
 - state representation in, 303
- Objectory, 30
- Objects, 10, 125–126
 - attributes for, 132
 - classes for, 134–135, 170
 - in concurrency, 420
 - constructing and destructing, 148–150, 247–248
 - diagrams for, 178–179
 - encapsulation of, 128–130
 - identical and equivalent, 528
 - messaging for, 130–131
 - notation for, 131–132
 - properties of, 127–129

- reactive, 439
- relationships with classes. *See* Relationships
- scope of, 147
- summary, 151–154
- tokens for, 288–289
- in use case realizations, 247–248
- Oblique paths for links, 180
- OCL (Object Constraint Language)
 - in activity diagrams, 536–537
 - benefits of, 497–498
 - expressions in. *See* Expressions in OCL
 - features of, 497
 - inherited associations in, 542–544
 - in interaction diagrams, 535–536
 - navigation in, 522–526
 - to and from association classes, 540–541
 - across associations, 523–526
 - within contextual instances, 522–523
 - through qualified associations, 541
 - package context and pathnames in, 500–501
 - in state machines, 537–540
 - summary, 546–550
- OclAny type, 505–506
- OclMessage expressions, 505, 544–546
- OclState type, 505
- OclType type, 505–506
- OclVoid type, 505
- Omnipotent analysis classes, 162
- OMT (Object Modeling Technique), 5–6
- one operator, 519
- One-to-many associations, 371
- One-to-one associations, 369–370
- OO flowcharts. *See* Activity diagrams
- Operands, 256–257
- Operations, 127
 - advanced syntax for, 145
 - in behaviors, 128–129
 - call action nodes for, 294–296
 - for classes, 142–146
 - analysis, 159
 - dependencies with, 196
 - design, 345–348
 - inheritance in, 210–211
 - for collections, 371–372, 513
 - access, 515–516
 - comparison, 514
 - conversion, 514
 - query, 514–515
 - selection, 516–518
 - of components, 400
 - in interfaces, 390
 - iteration, 518–522
 - overriding, 209
 - parameters in
 - default values, 145
 - direction, 143–145
 - lists, 142–143
 - precedence rules for, 504–505
 - query, 145–146
 - reusable, 404
 - scope of, 147–148
 - in use case realizations, 256–263
- opt operator, 257–261
- Optimizations in design classes, 348
- or operator, 507
- {ordered} property, 373–374
- OrderedSets, 374
- Ordering of object nodes, 303
- Organizational units, 291
- Organizing requirements, 58–59
- Orthogonal composite states, 462–465
- Orthogonal paths for links, 180
- out parameter, 144
- Output effects in activity diagrams, 319–320
- Outputs for interactions, 277–278
- {overlapping} constraint, 217
- Overmodeling collection classes, 373
- Overriding inheritance
 - process, 208–209
 - in use case generalization, 99–100
- Overview diagrams, interaction, 323–325
- Ownership in aggregation, 365

584 Index

- Package context, 500–501
- Packages, 223
 - architectural analysis, 231–235
 - dependencies, 200, 228–230, 234–235
 - features of, 224–226
 - finding, 232–234
 - generalization, 231
 - for namespaces, 226–227
 - nested, 227–228
 - summary, 234–237
 - visibility of, 138–139
- Paperwork, classes for, 170
- par operator
 - in concurrency, 423–425
 - semantics of, 257
- Parallel expansion nodes, 314
- Parallel working, packages for, 224
- «parameter» dependencies, 197–198
- Parameters
 - in activity diagrams, 321–322
 - in constructors, 148–149
 - in interaction occurrences, 276–277
 - in object nodes, 304–306
 - in operations
 - default values, 145
 - direction, 143–145
 - lists, 142–143
 - templates for, 355–356
- Parent actors, 98
- Parent use cases, 99–102
- Parentheses () for precedence, 505
- Partitions
 - activity, 290–293
 - class, 219
- Pathnames
 - in OCL, 500–501
 - in packages, 226
- Paths for links, 179–180
- Patterns
 - archetype, 170–171
 - facade, 406–407
- Performance in design classes, 348
- Permission dependencies, 200
- «permit» stereotype, 200
- Petri Nets, 284
- Phases in unified process
 - construction, 37–39, 41–43
 - elaboration, 37–41
 - inception, 37–40
 - transition, 37–39, 43–44
- Physical objects, classes for, 170
- Pins in object nodes, 305–306
- Platform-independent models (PIMs), 8
- Platform-specific models (PSMs), 8
- Plug-in algorithms, 405
- Plus signs (+) for visibility, 138
- Points in time, action nodes for, 297
- Polymorphism, 205–206
 - example, 213–215
 - features of, 211–212
 - inheritance in, 351
 - in interface realization, 394
 - summary, 221–222
- Ports for interfaces, 398–399
- Positive edge triggered change events, 452
- post: expression, 503, 529–530
- Postconditions
 - in OCL, 503, 529–530
 - in specifications, 80–81
 - in use case generalization, 99
- Powertypes, 218–220
- pre: expression, 503, 529–530
- @pre expression, 529–530
- Precedence rules, 504–505
- Preconditions
 - in OCL, 503, 529–530
 - in specifications, 80–81
 - in use case generalization, 99
- prepend operator, 518
- Primary actors, 80
- Primary scenarios, 81
- Primitive types
 - for attributes, 368
 - in OCL, 140, 506–509

- Primitiveness in design classes, 348–349
- Prioritize use cases, 54
- Private visibility
 - of analysis packages, 225
 - semantics of, 138–139
- Problem domains, design classes from, 344
- «process» stereotype, 402
- Process views, 23
- product operator, 517
- Profiles
 - Java, 489
 - UML, 22
- Project glossaries, 75–77
- Properties
 - of objects, 127–128
 - in OCL navigation, 522
- Protected visibility, 138–139
- Protocol state machines, 440
- Protocols in interfaces, 390, 398
- Prototype user interfaces, 54
- Provided interfaces, 391–393
- Pseudo-attributes, 191
- Pseudo-states for transition connections, 448
- PSMs (platform-specific models), 8
- Public visibility
 - of analysis packages, 225
 - semantics of, 138–139
- Python language
 - return values in, 145
 - template support for, 356
 - for use case editor, 433–434
- Qualified associations
 - features of, 193–194
 - navigation through, 541
- Qualified names, 226–227
- Quantifiers, universal, 62
- Query operations, 145–146
 - for collections, 514–515
 - in OCL, 503, 506
- Questionnaires for requirements, 63–64
- Questions in requirements interviews, 63
- Rational Objectory Process (ROP), 31
- Rational Rose tool, 56
- Rational Unified Process (RUP), 32–33
- Reactive objects, 439
- Real-Time Studio tool, 441
- Real-time systems, timing diagrams for, 427–430
- Real type
 - semantics of, 140
 - working with, 508–509
- ref operator, 257
- «refine» dependencies, 199
- Reflexive aggregation relationships, 366
- Reflexive associations, 185–186
- Regions
 - interruptible, 311–312
 - for submachines, 458–460
- Reified relationships, 375
 - association classes, 377–378
 - bidirectional associations, 376–377
 - many-to-many associations, 375–376
- Reitmann, Rich, 31
- reject operator, 520
- Relationships, 12, 175–176, 361–362
 - aggregation in
 - and composition, 363–364
 - semantics for, 364–367
 - artifact trace, 335
 - associations in. *See also* Associations
 - many-to-many, 375–376
 - many-to-one, 370–371
 - one-to-many, 371
 - one-to-one, 369–370
 - collections, 371–374
 - components, 400
 - composition in
 - and aggregation, 363–364
 - semantics for, 367–368
 - with structured classes, 378–382
 - dependencies, 195–200
 - in design, 335, 363
 - features of, 177

586 Index

- Relationships, *continued*
in generalization, 206–207
links, 177–180
in packages, 234–235
refining, 368–369
reified, 375
 association classes, 377–378
 bidirectional associations, 376–377
 many-to-many associations, 375–376
for states, 443
in structured classifiers, 378
summary, 201–203, 382–386
transitivity of, 230
in use case generalization, 99
- Remainder operation, 509
- repeat loops in sequence diagrams, 263
- Repetition in main flow, 83–84
- Required interfaces, 391–393
- Requirements
attributes of, 59–60
elicitation of, 61–62
finding, 61–63
functional and non-functional, 57–58
interviews for, 63
in iteration workflow, 36
models for, 56
 in use case modeling, 70
 in use case realization design, 416
organizing, 58–59
questionnaires for, 63–64
tracing, 90–91
well-formed, 56–57
workflow, 49–50
 defining, 55–60
 detail, 53–54
 features of, 51–52
 importance of, 55
 software, 52–53
 summary, 65–66
workshops for, 64
- Requirements engineering, 51
- RequisitePro tool, 56, 90
- Responsibilities
in analysis classes, 161–162
in CRC analysis, 165–167
- result operation, 544
- Return messages in use case realizations,
246–247
- return parameter, 144
- Return types, 142–143
- Reuse
of operations, 404
submachines for, 465
templates for, 355
- Risk attribute, 60
- Roles, 32–33
activity partitions for, 291
for associations, 181–182
in structured classifiers, 378, 381
- ROP (Rational Objectory Process), 31
- round operation, 509
- Royce, Walker, 31
- RUP (Rational Unified Process), 32–33
- RUP stereotypes, finding classes by, 167–169
- Scenarios in specifications, 81
- Scope
for access, 147–148
of interaction occurrences, 276
- «script» stereotype, 488
- Scripts for sequence diagrams, 253–254
- SDL (Specification and Description Language),
30
- Secondary actors, 80
- Secondary scenarios, 81
- Security systems, concurrency for, 420–422
- Segments, multiple insertion, 108–109
- Selection behaviors of object nodes, 303
- Selection operations
for collections, 516–518
for iterators, 520
- «selection» stereotype, 303, 319
- Selectors for lifelines, 245
- Self-delegation in sequence diagrams, 252

- self keyword
 - for context, 501
 - in OCL navigation, 523
- Semantic backplanes, 16
- Semantic boundaries, packages for, 224
- Semantics
 - activity, 288–289
 - aggregation, 364–367
 - composition, 367–368
- Semicolons (;) for actions, 449
- «send» stereotype, 198
- Sending signals in activity diagrams, 314–317
- SEP (software engineering process), 28
- seq operator, 257
- Sequence diagrams
 - concurrency in, 422–425
 - interaction occurrences in, 274–276
 - in use case realizations, 249
 - activation in, 253
 - documenting, 253–254
 - lifelines and messages in, 249–252
 - state invariants and constraints in, 254–256
- Sequence numbers in communication diagrams, 425–426
- Sequences, 374, 512–513
- «service» stereotype, 402
- Services
 - and encapsulation, 130
 - interfaces for, 391
- Sets, 374
 - generalization, 216–218
 - of object nodes, 303
- Shallow history in state machines, 468–469
- Should have requirements, 59
- Side effects
 - in OCL expressions, 497
 - in query operations, 145
- Signal classifiers, 19
- Signal events, 450–451
- «signal» stereotype, 314
- Signals
 - in activity diagrams, 314–317
 - for messages, 246
- Signatures
 - in interface operations, 390
 - message, 130–131
 - for operations, 142–143
 - in overriding, 209
 - in polymorphism, 212, 214
- Simple composite states, 460–462
- Simple deviations, 81
- Simplicity
 - in analysis models, 123
 - in analysis packages, 233
 - in use cases, 111
- Size
 - analysis classes, 162
 - use cases, 111
- size operator
 - for collection queries, 515
 - for strings, 509
- Slashes (/) for comments, 504
- Smalltalk language
 - return types in, 143
 - template support for, 356
- Software engineering process (SEP), 28
- Software requirements specification (SRS), 53
- Software requirements workflow, 52–53
- Solution domains, design classes from, 344
- sortedBy operator, 520
- Source nodes with tokens, 289
- «source» stereotype, 488
- Spaghetti code, 350
- Specialization, 207
- Specification and Description Language (SDL), 30
- «specification» stereotype, 402
- Specifications, 15–17, 78–79
 - actors in, 80
 - alternative flows in, 85–90
 - for artifacts, 486
 - in behaviors, 128
 - brief descriptions in, 80

588 Index

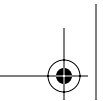
- Specifications, *continued*
vs. implementations, 389–391
in interfaces, 390–391
main flow in, 81–85
preconditions and postconditions in,
80–81
use case IDs, 80
use case names, 79–80
- SRS (software requirements specification), 53
- Stability attribute, 60
- Stakeholders
activity diagrams for, 286
in analysis models, 123
in requirements, 51
- Stand-alone style pins, 306
- Standard libraries in Java, 393, 407
- State dependent behavior, 128
- State invariants in sequence diagrams,
254–256
- State machines, 437–438
advanced, 457–458
behavioral and protocol, 440
for class models, 440–441
diagrams for, 442–443
events in, 439–440, 448–449
call, 449
change, 452–453
signal, 450–451
time, 453
transition, 446
features of, 439
history, 468–470
OCL in, 537–540
states in. *See* States
submachine communication, 467
summary, 453–455, 470–471
transitions for, 439–440
branching, 448
connecting, 447
features of, 445–447
in UP, 441
- State property, 127–129
- States, 439–440, 443–444
composite
features of, 458–460
orthogonal, 462–465
simple, 460–462
in object nodes, 303
submachine, 465–466
syntax, 444–445
in timing diagrams, 427–429
- Status attribute, 60
- Step in main flow feature, 99
- Stereotypes, 20–21, 134–135
«access» 200, 229–230
«artifact» 487–488
«bind» 355–356
«boundary» 167–168
«buildComponent» 402
«call» 197
«centralBuffer» 303
«control» 168–169
«create» 248
«decisionInput» 298
«deployment» 488
«derive» 199
«destroy» 248
«device» 484
«directory» 489
«document» 488
«EJB» 489
«entity» 169, 402
«executable» 488
«execution environment» 484
«extend» 105–107
«external» 292
«file» 488
«framework» 226
«implementation» 402
«import» 200, 229
«include» 102–105
«instantiate» 134–135, 198
«JavaClassFile» 489
«JavaSourceFile» 489

- «library» 488
- «manifest» 478
- «merge» 230
- «modelLibrary» 226
- «parameter» 197–198
- «permit» 200
- «process» 402
- «refine» 199
- «script» 488
- «selection» 303, 319
- «send» 198
- «service» 402
- «signal» 314
- «source» 488
- «specification» 402
- «substitute» 198
- «subsystem» 402–403
- «topLevel» 224
- «trace» 198, 229
- «transformation» 320
- «use» 196–197, 228–229
- for analysis packages, 226
- class, 146, 160
- component, 402
- finding classes by, 167–169
- in interfaces, 390
- for packages, 224
- Stream expansion nodes, 314
- Streaming in activity diagrams, 317–318
- strict operator, 258
- String type
 - semantics of, 140
 - working with, 508, 510
- Structure, 10–11, 37–39
- Structured classes, 378–382
- Structured classifiers, 378–379, 400
- Subclasses for invariants, 527
- Subjects in use case modeling, 71
- Submachines, 458–459
 - communication with, 467
 - in orthogonal composite states, 462
 - states in, 465–466
 - synchronization of, 462–464, 467
- subOrderedSet operator, 518
- subSequence operator, 518
- Substates, 460
- Substitutability principle, 394
- «substitute» dependencies, 198
- substring operator, 510
- «subsystem» stereotype, 402–403
- Subsystems, 403
 - designing, 389
 - interactions, 426–427
- Sufficiency in design classes, 347–348
- Sufficiently complete models, 17
- sum operator, 515
- SUMR toolset
 - for candidate requirements, 91
 - use case editor example, 430–435
- Superstates
 - benefits of, 460
 - history for, 468–469
- symmetricDifference operator, 517
- Synchronization of submachines, 462–464, 467
- Synchronous messages in use case realizations, 246–247
- Synonyms in project glossaries, 76
- System boundaries in use case modeling, 71
- Tagged values
 - for analysis classes, 160
 - features of, 21–22
 - in interfaces, 390
- Target nodes with tokens, 289
- TargetRelease attribute, 60
- Taxonomies, 58
- Templates
 - classes as, 135
 - collections as, 512–513
 - for design classes, 354–356
- Testing
 - in iteration workflow, 36
 - state machines, 441

590 Index

- Textures in UML models, 20
- Things, 12
- Time
 - in activity diagrams, 296–297
 - as actor, 73
 - in sequence diagrams, 251
 - in state machines, 453
- Timing diagrams, 427–430
- toInteger operator, 509
- Token games, 289
- toLower operator, 509
- «topLevel» stereotype, 224
- toReal operator, 509
- toUpper operator, 509
- Trace relationships, artifact, 335
- «trace» stereotype
 - dependencies, 198
 - for packages, 229
 - in RUP, 32–33
 - in use case realization–design, 416
- Tracing requirements, 90–91
- Traffic cases, 30
- «transformation» condition, 320
- Transition phase, 37–39, 43–44
- Transitions in state machines, 439–440
 - branching, 448
 - connecting, 447
 - features of, 445–447
- Transitive aggregation, 365
- Transitive composition, 367
- Transitivity in dependencies, 230
- Triggers
 - for alternative flows, 88
 - for events, 316, 452
- Tuples, 509–510
- Types
 - for attributes, 368
 - for lifelines, 244
 - notation for, 140
 - in OCL, 506–509
 - in structured classifiers, 378
 - for tuples, 510
- UML basics, 3–5
 - architecture, 23–24
 - building blocks, 11–15
 - common mechanisms, 15–16
 - adornments, 17–18
 - divisions, 18–19
 - extensibility, 19–22
 - specifications, 15–17
 - development of, 5–7
 - and MDA, 7–9
 - objects in, 10
 - structure of, 10–11
 - summary, 24–26
 - unification in, 9–10
- Unidirectional links, 178
- Unified process, 27–28
 - activities
 - in use case detailing, 77–78
 - in use case modeling, 69–77
 - activity diagrams in, 285–286
 - axioms, 34–35
 - development of, 29–32
 - features of, 28–29
 - instantiating, 34
 - iterative and incremental processes, 35–37
 - phases in
 - construction, 37–39, 41–43
 - elaboration, 37–41
 - inception, 37–40
 - transition, 37–39, 43–44
 - and RUP, 32–33
 - structure, 37–39
 - summary, 44–45
- Unified Software Development Process (USDP), 28
- union operator, 516
- {unique} property, 373–374
- Units, packages for, 224
- Universal quantifiers, 62
- UnlimitedNatural type, 140
- {unordered} property, 373–374

- Upper bounds of object nodes, 302
- UpperCamelCase naming convention
 - for classes, 137
 - for interfaces, 392
- Usage dependencies, 196–198
- USDP (Unified Software Development Process), 28
- Use case classifiers, 19
- Use case modeling, 67–68
 - actions in, 111
 - advanced, 95–97, 110
 - applying, 91–92
 - extends in, 105–109
 - features of, 69
 - functional decomposition in, 112–113
 - generalization in
 - actor, 97–99
 - use case, 99–102
 - hints and tips for, 111–113
 - includes in, 102–105
 - project glossaries in, 75–77
 - requirements tracing in, 90–91
 - size and simplicity of, 111
 - specifications. *See* Specifications
 - summary, 92–94, 113–115
 - unified process activities
 - in use case detailing, 77–78
 - in use case modeling, 69–77
 - in use case realization–design, 416
- Use case realization–design, 413–414
 - concurrency modeling, 419–420
 - active classes in, 420–422
 - in communication diagrams, 425–426
 - in sequence diagrams, 422–425
 - consequences of, 415–416
 - example, 430–435
 - interaction diagrams in, 417–419
 - parts of, 416
 - subsystem interactions, 426–427
 - summary, 436
 - timing diagrams, 427–430
- Use case realizations–analysis, 239–242
 - advanced, 273
 - in analysis packages, 224
 - combined fragments and operators in, 256–263
 - communication diagrams in, 264–268
 - continuations in, 279–281
 - elements of, 243–244
 - features of, 242
 - interactions in, 244
 - diagrams, 243–244, 248–249
 - occurrences, 274–276
 - lifelines in, 244–245
 - messages in, 246–248
 - sequence diagrams in, 249–256
 - summary, 268–271, 281–282
- Use cases
 - activity partitions for, 291
 - in analysis packages, 224
 - analyzing. *See* Analysis classes
 - detailing, 77–78
 - diagrams for, 75
 - editor example, 430–435
 - identifying, 74–75, 80
 - names for, 79–80
 - in requirements workflow, 54
 - in unified process activities, 77–78
 - views, 23
- «use» stereotype, 196–197, 228–229
- User interface boundary classes, 167–168
- Utility classes for attributes, 368
- Variables, 531–533
- Vector class, 372
- Views, 23. *See also* Diagrams
- Visibility
 - for analysis classes, 160
 - of analysis packages, 225
 - of ports, 398–399
 - working with, 138–139



592 Index

Visual Basic language
interface names in, 392
template support for, 356

Want to have requirements, 59

Well-formed design classes,
347–349

Well-formed requirements, 56–57

When keyword, 453

While loops

in main flow, 85

in sequence diagrams, 263

Windows. *See* Diagrams

Workers, 32–33

Workflow

analysis. *See* Analysis workflow

design. *See* Design and design workflow

implementation. *See* Implementation

iteration, 36–37

requirements. *See* Requirements

Workshops for requirements, 64

WxPython library, 433–434

Xor operator, 507

