
Index

A

- Abstract classes, 17, 80
- Actors (UML), 51
- Adapter design pattern, 340, 341
- Adaptive Object Model, 353
- Aftereffect guarantees, object contracts, 7–8
- Applications
 - application-specific objects, 10–12
 - Application objects, Page-Jones domain divisions, 135
 - architecture (*See* Architecture of applications) definition, 3
 - policies, design description, 57
- Architectural objects, Page-Jones domain divisions, 135
- Architecture of applications
 - basics, 27–28
 - object collaborations, building models, 192
 - object collaborations, influences on, 172–173
- Architecture of applications, control styles. *See also* Architecture of applications, styles; Controllers, object role stereotype
 - centralized, 30, 197, 198–200
 - centralized, advantages/disadvantages, 198, 201–203
 - control centers, 196, 205
 - control centers, designing for similar systems, 230–236
 - delegated, 31–32, 33, 197, 200–201
 - delegated, advantages/disadvantages, 198, 201–203
 - dispersed, 30–31, 197
 - dispersed, advantages/disadvantages, 198, 203, 204–205
 - Guesser object/dictionaries neighborhood, 225–229
 - MessageBuilder object event, 205
 - MessageBuilder object event, basics, 206–208
 - MessageBuilder object event, centralizing control, 208–220
 - MessageBuilder object event, decision making, moving responsibilities, 224–225
 - MessageBuilder object event, decision making, refactoring into state methods, 220–221
 - MessageBuilder object event, final design, 229–230
 - object collaborations, 155
 - overview, 196
- Architecture of applications, styles. *See also* Architecture of applications, control styles
 - blackboard, 28

Index

Architecture of applications, *(continued)*

- layered, 28, 29, 32–34
- layered, locating objects, 34–36
- pipes-and-filter, 28, 30

B

- Blackboard architectural style, 28
- Builder design pattern, object collaborations, 171
- Business objects, Page-Jones domain divisions, 135
- Business rules, design description, 57

C

Candidate objects

- characterizing in larger context, 98–99
- clustering/connecting, 99–101
- collaborators, 80
- conceptual objects, 58–60
- defending, 104–105
- descriptions, 93–98
- discarding, 103–104
- exploratory design stage, 60–61
- naming, 88–93
- responsibilities, 80
- reviewing, 105–106
- role stereotypes, 93–98
- roles, 79–80, 101–103
- search strategies, basics, 84–85
- search strategies, themes, 85–87
- steps in finding/assessing, 78–79
- transitioning to classes and interfaces, 80
- writing stories, 80–83
- writing stories, collaborations, 152–153, 154

Candidates, Responsibilities, Collaborators. *See* CRC cards

Case statements, 20

Classes

- abstract, 17, 80
- components, 18
- concrete, 80
- finding candidate objects, 80
- inheritance, 16–17
- instances, 13–16
- libraries of classes in frameworks, 25

- superclasses and subclasses, 16–17
- Collaboration of objects. *See* Object collaborations
- Collaborators, finding candidate objects, 80. *See also* Object collaborations
- Command pattern, 64–66
- Components, 18
- Composite design pattern, object collaborations, 171
- Composition, object models, 16
- Conceptual objects, 58–60
 - judging merit, 60–61
 - running collaboration simulations, 182
- Concrete classes, 80
- Concrete method, 330
- Conditions-of-use guarantees, object contracts, 7–8
 - collaborations, 156
- Consequences, pattern elements
 - definition, 20
 - Double Dispatch pattern, 24
- Consistency in design, 73–74
- Context, pattern elements
 - definition, 20
 - Double Dispatch pattern, 24
- Contracts, object collaborations, 156, 307
 - basics, 7–8
 - contractual relations, 308
 - definition, 3
 - obligations and benefits, 309–310
 - preconditions and postconditions, 308–309
- Control styles, application architecture. *See also* Architecture of applications, styles; Controllers, object role stereotype
 - centralized, 30, 197, 198–200
 - centralized, advantages/disadvantages, 198, 201–203
 - control centers, 196, 205
 - control centers, designing for similar systems, 230–236
 - delegated, 31–32, 33, 197, 200–201
 - delegated, advantages/disadvantages, 198, 201–203
 - dispersed, 30–31, 197
 - dispersed, advantages/disadvantages, 198, 203, 204–205
 - Guesser object/dictionaries neighborhood, 225–229

- MessageBuilder object event, 205
 - MessageBuilder object event, basics, 206–208
 - MessageBuilder object event, centralizing control, 208–220
 - MessageBuilder object event, decision making, moving responsibilities, 224–225
 - MessageBuilder object event, decision making, refactoring into state methods, 220–221
 - MessageBuilder object event, final design, 229–230
 - object collaborations, 155
 - overview, 196
 - Controllers, object role stereotype, 4. *See also*
 - Control styles, application architecture
 - candidate descriptions, 93–94
 - collaborations, simulating, 179
 - in layered style applications, 34–35
 - object collaborations, 163
 - object collaborations, *versus* coordinators, 164
 - Conversations, design description, 54–55, 56
 - Coordinators, object role stereotype, 4
 - candidate descriptions, 93–94
 - in layered style applications, 34–35
 - versus* mediators, 229
 - object collaborations, 163, 164
 - CRC cards
 - candidate objects, collaborations, recording, 151–152
 - candidate objects, connecting/clustering cards, 99
 - candidate objects, defining, 93–94
 - candidate objects, filling out, 100
 - candidate objects, finding patterns, 100–101
 - candidate objects, information required, 61–62, 63, 67
 - object collaborations, running simulations, 182
 - origin, 36
 - recording object responsibilities, 122–123
-
- D**
- Descriptions (design)
 - analysis, 49–50
 - application policies, 57
 - basics, 36
 - business rules, 57
 - design notes, 57
 - exceptions, 56–57
 - object responsibilities, 131–132
 - Responsibility-Driven Design, 44–47
 - UML (Unified Modeling Language), 241
 - usage, 50–51
 - usage, conversations, 54–55, 56
 - usage, scenarios, 53–54, 56
 - usage, use cases, 51–53, 56
 - Design patterns
 - Adapter, 340, 341
 - basics, 18–19
 - benefits to developers, 20, 25
 - Command pattern, 64–66
 - delegation technique, 341–342
 - Double Dispatch pattern, 20–25, 175, 176, 177
 - increasing flexibility, 340–342
 - Mediator, 339
 - object collaborations, 170–171
 - object collaborations, building models, 192
 - State, 341
 - Strategy, 337–338
 - Template Method, 330–331
 - Design process. *See also* Descriptions (design)
 - agile development practices, 373
 - analysis, 45–47
 - basics, 40–42
 - collaboration objects, responsibilities and control styles, 70–71
 - conceptual objects, 58–60
 - conceptual objects, judging merit, 60–61
 - connection problems, 358, 363–364
 - consistency, 73–74
 - control problems, 358, 362–363, 364–365
 - core problems, 356, 357–358
 - CRC cards, 61–62, 63, 67
 - design decisions, guidelines, 67–68
 - design decisions, testing designs with details, 68, 70
 - “fudging”, 371–374
 - general problems, 370–371
 - problem frames, 358–361
 - problems, 356
 - responsible design, 371–374
 - revealing problems, 356, 361
 - revealing problems, and wicked problems, 365–366
 - revealing problems, redefining before solving, 368

Index

- Design process, (*continued*)
 revealing problems, solving, 366–368
 revealing problems, synthesizing before solving, 369
 transformation problems, 359
 workpiece problems, 359
 Design reviews for reliable collaborations, 311–312
 Dialogs. *See* Conversations
 Domain objects
 basics, 8–10
 object collaborations, building models, 192
 object responsibility restrictions, 135–136
 Page-Jones domain divisions, 135
 Double Dispatch design pattern, 20–25
 object collaborations, 175, 176, 177
-
- E**
- Errors/exceptions
 basics, 288–294
 definition, 287
 design, limiting scope, 300–302
 design, listing possibilities, 299–300
 design description, 56–57
 handling, 294–296
 handling, documenting designs, 303–307
 handling, recording policies, 302–303
 of objects *versus* use cases, 288
 responsibilities, 296–299
 Events (objects)
 collaborations, 169–170
 collaborations, building models, 192
 collaborations, running simulations, 180–182
 sources of object responsibilities, 123–124
 Exceptions. *See* Errors/exceptions
 External interfacers, object role stereotype
 candidate descriptions, 93
 collaboration identification, 165–166
-
- F**
- Facade design pattern, object collaborations, 171, 173, 174, 175
 Factory method, 330
 Flexibility, 71–72
 Flexibility in design, 71–72
 documentation, audience considerations, 344–345, 347, 348
 documentation, descriptions of variations, 345, 347–350
 documentation, UML-F (Unified Modeling Language, frameworks), 342–344, 345, 346
 documentation, UML (Unified Modeling Language), 345, 347
 documentation, 342–344, 345
 variations, describing for documentation, 345, 347–350
 variations, working into existing software, 350–352
 Flexible design
 advantages/disadvantages, 319–320
 decisions, objects needing flexibility, 320–324
 degrees of flexibility, 317–319
 overview, 316–317
 patterns, Adapter, 340, 341
 patterns, delegation technique, 341–342
 patterns, Mediator, 339
 patterns, State, 341
 patterns, Strategy, 337–338
 patterns, Template Method, 330–331
 patterns, ways to increase flexibility, 340–342
 Responsibility-Driven Design, 71–72
 variations, creating knobs for developers to turn, 337–338
 variations, hot spots, recording on cards, 324–327
 variations, hot spots, solving, 327–328
 variations, inserting design placeholders, 335–337
 variations, placing variable information into information holders, 334–335
 variations, strategies for realizing variations, 329–330
 variations, supporting with template and hook methods, 330–333
 variations, times needed, 333–334
 Flyweight design pattern, object collaborations, 171
 Forces, pattern elements
 definition, 19
 Double Dispatch pattern, 24
 Foundation objects, Page-Jones domain divisions, 135
 Framelets, 353

Frameworks

- advantages/disadvantages to developers, 26–27
- basics, 25–26
- control styles, 201
- UML-F (Unified Modeling Language, frameworks), 342–344, 345, 346
- Fundamental objects, Page-Jones domain divisions, 135

G–H

- Glossaries, 58
- Hook method, 330
- Hot spots, variations in design
 - cards, 71, 72
 - cards, recording variations, 324–327
 - solving, 327–328

I–K

- Information holders, object role stereotype, 4
 - candidate descriptions, 93
 - in layered style applications, 34–35
 - object collaborations, 159–160
 - variable information in flexible designs, 334–335
- Inheritance, classes, superclasses, and subclasses, 16–17
- Instances of classes, 13–16
 - inheritance, 16–17
- Integrative and incremental processes, 42–43
- Interfacers, object role stereotype, 4
 - candidate descriptions, 93
 - in layered style applications, 34–35
 - object collaborations, 164–166
- Interfaces
 - basics, 12
 - finding candidate objects, 80
- Internal interfacers, object role stereotype
 - candidate descriptions, 93
 - collaboration identification, 165

L–M

- Layered architectural style, locating objects, 34–36

- Libraries of classes in frameworks, 25

- Mediator design pattern, 339
 - object collaborations, 171
- Model-View-Controller roles, assigning responsibilities of objects, 129–130, 131
- Multiple stakeholder perspectives, 49, 50, 71

N

- Names, pattern elements
 - definition, 19
 - Double Dispatch pattern, 23
- Naming objects, 88–93
- Narratives. *See* Stories
- Neighborhood of objects, 17
- Neighborhoods of objects
 - collaborations, 151

O

- Object collaborations
 - architecture's influences, 172–173
 - basics, 150
 - control styles, 70–71, 155
 - definition, 3
 - degree of trust, 155–157
 - design, based on use cases or events, 169–170
 - design, patterns, 170–171
 - design stories, 152–153, 154
 - feasibility of collaborations, 187–188
 - guidelines for design, 183–184
 - guidelines for design, exceptional conditions, 190–191
 - guidelines for design, Law of Demeter case study, 184–187
 - guidelines for making connections, 188–190
 - identification strategies, 158–159
 - neighborhoods, 17
 - preparations, 150–151
 - raw materials for model building, 192
 - recording candidates on CRD cards, 151–152
 - roles-responsibilities-collaborations model, 5–7
 - subsystems, 17
 - troubleshooting problems, 173–176

Index

- Object collaborations, reliability
 - contracts, 307
 - contracts, contractual relations, 308
 - contracts, obligations and benefits, 309–310
 - contracts, preconditions and postconditions, 308–309
 - design reviews, 311–312
 - errors/exceptions, basics, 288–294
 - errors/exceptions, definition, 287
 - errors/exceptions, design, limiting scope, 300–302
 - errors/exceptions, design, listing possibilities, 299–300
 - errors/exceptions, handling, 294–296
 - errors/exceptions, handling, documenting designs, 303–307
 - errors/exceptions, handling, recording policies, 302–303
 - errors/exceptions, of objects *versus* use cases, 288
 - errors/exceptions, responsibilities, 296–299
 - failures, consequences, 278–279
 - information from use cases, 286
 - overview, 285–286
 - system reliability, 280
 - trust regions, 280
 - trust regions, decisions on placement of responsibilities, 284–285
 - trusted collaborations, 280–281
 - trusted collaborations, *versus* untrusted, 281–284
- Object collaborations, responsibilities
 - connecting objects, 166–167
 - connecting responsibilities, 151–152
 - control styles, 70–71
 - factor in frequency of objects, 153–155
 - subresponsibilities, 168–169
- Object collaborations, role stereotypes
 - controllers, 163
 - controllers *versus* coordinators, 164
 - coordinators, 163, 164
 - information holders, 159–160
 - interfacers, 164–166
 - service providers, 162
 - structurers, 160–162
- Object collaborations, simulations
 - basics, 176–177
 - goal setting, 178
 - planning, 177–180
 - running, 180–182
- Object collaborations, stories
 - basics, 240
 - description guidelines, 264–270
 - development strategies, 241
 - final stories, 273–274
 - limitations of UML diagrams, 258–262
 - listing items to cover, 243
 - organization basics, 270–273
 - preserving stories, 274
 - scope, depth, and tone, 242–243
 - selecting forms best-suited for stories, 263–264
 - views, bird's eye, 244–245
 - views, collaborators only, 245–250
 - views, focused interactions among collaborators, 253–254
 - views, implementations, 254
 - views, in-depth, 250–253
 - views, sequences of interactions among collaborators, 250
- Object models
 - composition relationship, 16
 - inheritance relationship, 16–17
- Object role stereotypes
 - candidate descriptions, 93–94
 - controllers, 4, 34–35
 - coordinators, 4, 34–35
 - information holders, 4, 34–35
 - interfacers, 4, 34–35
 - service providers, 4, 34–35
 - sources of object responsibilities, 121
 - structurers, 4, 34–35
- Object roles, 3–4
 - candidate objects, 79–80, 101–103
 - collaborations, simulating, 178
 - implementing responsibilities, 141–143
 - roles-responsibilities-collaborations model, 5–7
- Objects
 - application-specific, 10–12
 - candidates (*See* Candidate objects)
 - class components, 18
 - classes, instances, 13–16
 - contracts (*See* Contracts, object collaborations)
 - controllers (*See* Architecture of applications, control styles)
 - definition, 3
 - designer perspective, 11–12

domains, 8–10
 events (*See* Events (objects))
 interfaces, 12
 naming, 88–93
 patterns (*See* Design patterns)
 responsibilities (*See* Responsibilities of objects)
 software *versus* physical machinery, 2–3
 user perspective, 11–12
 Observer design pattern, object collaborations,
 171

P–Q

Packages (UML), 244–245
 Page-Jones domain divisions, 135
 Patterns (design)
 Adapter, 340, 341
 basics, 18–19
 benefits to developers, 20, 25
 Command pattern, 64–66
 delegation technique, 341–342
 Double Dispatch pattern, 20–25, 175, 176, 177
 increasing flexibility, 340–342
 Mediator, 339
 object collaborations, 170–171
 object collaborations, building models, 192
 State, 341
 Strategy, 337–338
 Template Method, 330–331
 Pipes-and-filters architectural style, 28, 30
 Predictability in design, 73–74
 Problems
 design process, 356
 design process, connections, 358, 363–364
 design process, controls, 358, 362–363, 364–365
 design process, core problems, 356, 357–358
 design process, general problems, 370–371
 design process, problem frames, 358–361
 design process, revealing problems, 356, 361
 design process, revealing problems, and wicked
 problems, 365–366
 design process, revealing problems, redefining
 before solving, 368
 design process, revealing problems, solving,
 366–368
 design process, revealing problems, synthesiz-
 ing before solving, 369

design process, transformations, 359
 design process, workpieces, 359
 pattern elements, definition, 19
 pattern elements, Double Dispatch pattern, 23
 Profiles, Unified Modeling Language, 342
 Project definition, 44
 Project planning, 44

R

Reliability in design, 73
 Responsibilities of objects
 collaborations, connecting, 151–152, 166–167
 collaborations, factor in frequency, 153–155
 collaborations, subresponsibilities, 168–169
 definition, 3
 finding candidate objects, 80
 overview, 110–111
 recording on CRC cards, 122–123
 roles-responsibilities-collaborations model, 5–7
 testing for well-formed objects, 145–146
 Responsibilities of objects, assignment strategies
 basics, 125–126
 coherent statements, 135
 distributing system intelligence, 133–134
 eliminating nonessential or overlapping respon-
 sibilities, 136–138
 general statements, 128–129
 initial assignments, 128
 judging ability of object to divide or share work,
 132
 keeping behaviors with related information, 133
 limiting scope, 133
 limiting sharing of information, 134–135
 Model-View-Controller roles, 129–130, 131
 POSA, 129–131
 recording on CRC cards, 126–128
 restricting to single domain, 135–136
 troubleshooting problems, 138–140
 varying description length, 129–131
 word choices, 131–132
 Responsibilities of objects, implementations,
 140–141
 designing methods supporting responsibilities,
 144
 implementation-specific responsibilities,
 124–125

Index

- Responsibilities of objects, sources
 - basics, 111–112
 - design stories, 117–119
 - implementation-specific responsibilities, 124–125
 - important object events, 123–124
 - object role stereotypes, 121
 - private responsibilities supporting public ones, 121–123
 - relationships between candidates, 123
 - system behaviors, 112–115
 - system behaviors, filling needs between system behaviors and use cases, 116–117
 - themes, 117–119
 - theoretical chains of reasoning, 119–120
 - use cases, 112–115
 - Responsibility-Driven Design
 - analysis, 45–47
 - basics, 40–42
 - collaboration objects, responsibilities and control styles, 70–71
 - conceptual objects, 58–60
 - conceptual objects, judging merit, 60–61
 - consistency, 73–74
 - CRC cards, 61–62, 63, 67
 - descriptions, 44–47
 - descriptions, analysis, 49–50
 - descriptions, application policies, 57
 - descriptions, business rules, 57
 - descriptions, design notes, 57
 - descriptions, exceptions, 56–57
 - descriptions, usage, 50–51
 - descriptions, usage, conversations, 54–55, 56
 - descriptions, usage, scenarios, 53–54, 56
 - descriptions, usage, stories, 52–53
 - descriptions, usage, stories for collaborations, 152–153, 154
 - descriptions, usage, use cases, 51–53, 56
 - design decisions, guidelines, 67–68
 - design decisions, testing designs with details, 68, 70
 - design patterns, 62, 64–67
 - flexibility, 71–72
 - glossaries, 58
 - interactive and incremental processes, 42–43
 - multiple stakeholder perspectives, 49, 50, 71
 - predictability, 73–74
 - project definition, 44
 - project planning, 44
 - reliability, 73
 - stages, exploratory, 47, 60–61
 - stages, refinement, 48, 70–71
 - Roles of objects, 3–4
 - candidate objects, 79–80, 101–103
 - collaborations, simulating, 178
 - definition, 3
 - finding candidate objects, 79–80
 - implementing responsibilities, 141–143
 - roles-responsibilities-collaborations model, 5–7
 - Roles of objects, stereotypes, 4–5
 - candidate descriptions, 93–94
 - in layered style applications, 34–35
 - sources of object responsibilities, 121
 - Roles-responsibilities-collaborations model, 5–7
-
- ## S
- Scenarios
 - design description, 53–54, 56
 - sources of object responsibilities, 112–115
 - Search strategies for candidate objects
 - basics, 84–85
 - steps in finding/assessing, 78–79
 - themes, 85–87
 - Semantic objects, Page-Jones domain divisions, 135
 - Service providers, object role stereotype, 4
 - candidate descriptions, 93–94
 - in layered style applications, 34–35
 - naming, 89
 - object collaborations, 162
 - Solutions, pattern elements
 - definition, 20
 - Double Dispatch pattern, 24
 - Sources of object responsibilities
 - basics, 111–112
 - design stories, 117–119
 - implementation-specific responsibilities, 124–125
 - object role stereotypes, 121
 - private responsibilities supporting public ones, 121–123
 - relationships between candidates, 123

- system behaviors, 112–115
 - system behaviors, filling needs between filling needs between and use cases, 116–117
 - themes, 117–119
 - theoretical chains of reasoning, 119–120
 - use cases, 112–115
 - Stages of design
 - exploratory, 47, 60–61
 - refinement, 48, 70–71
 - Stakeholder perspectives in design, 49, 50, 71
 - State design pattern, 341
 - object collaborations, 171
 - Stereotypes of object roles
 - candidate descriptions, 93–94
 - controllers, 4, 34–35, 163
 - controllers, *versus* coordinators, 164
 - coordinators, 4, 34–35, 164
 - information holders, 4, 34–35, 159–160
 - interfacers, 4, 34–35, 164–166
 - service providers, 4, 34–35, 162
 - sources of object responsibilities, 121
 - structurers, 4, 34–35, 160–162
 - Stories
 - design description, 52–53
 - design description, guidelines, 264–270
 - final stories, 273–274
 - finding candidate objects, 80–83
 - limitations of UML diagrams, 258–262
 - listing items to cover, 243
 - object collaborations, 152–153, 154
 - organization basics, 270–273
 - preserving stories, 274
 - selecting forms best-suited for stories, 263–264
 - sources of object responsibilities, 117–119
 - views, bird's eye, 244–245
 - views, collaborators only, 245–250
 - views, focused interactions among collaborators, 253–254
 - views, implementations, 254
 - views, in-depth, 250–253
 - views, sequences of interactions among collaborators, 250
 - Strategy design pattern, 337–338
 - object collaborations, 171
 - Structural objects, Page-Jones domain divisions, 135
 - Structurers, object role stereotype, 4
 - candidate descriptions, 93
 - in layered style applications, 34–35
 - object collaborations, 160–162
 - Subclasses, 16–17
 - Subsystems of objects, 17
 - Superclasses, 16–17
 - Switch statements, 20
 - System behaviors, sources of object responsibilities, 112–117
-
- T**
- Template method, 330–331, 332
 - Template Method design pattern, 330–331
 - Themes
 - object collaborations, building models, 192
 - sources of object responsibilities, 117–119
 - Troubleshooting problems
 - object collaborations, 173–176
 - object responsibilities, assignment strategies, 138–140
 - Trust regions, 280
 - decisions on placement of responsibilities, 284–285
 - Trusted collaborations, 280–281
 - versus* untrusted, 281–284
-
- U**
- UML-F (Unified Modeling Language, frameworks), 342–344, 345, 346
 - UML (Unified Modeling Language)
 - actors, 51
 - design descriptions, 36
 - documentation, flexible design, 345, 347
 - exception handling, 305–307
 - for frameworks (*See* UML-F (Unified Modeling Language, frameworks))
 - packages, 244–245
 - profiles, 342
 - relationships between candidate objects, 123
 - stories, 241
 - stories, limitations of UML diagrams, 258–262
 - stories, views, bird's eye, 244–245
 - stories, views, collaborators only, 245–250
 - stories, views, focused interactions among collaborators, 253–254

Index

UML (Unified Modeling Language), (*continued*)
 stories, views, implementations, 254
 stories, views, in-depth, 250–253
 stories, views, sequences of interactions among
 collaborators, 250
 systems and subsystems, 244–245
Unified Modeling Language. *See* UML (Unified Mod-
eling Language)
Use cases
 collaborations, building models, 192
 collaborations, simulating, 179
 conversation form, 54–55
 design description, 51–53, 56
 essential, 75
 narrative form, 52–53
 object collaborations, 169–170
 scenario form, 53–54
 sources of object responsibilities, 112–115,
 116–117
 Use case map, 276

User interfacers, object role stereotype
 candidate descriptions, 93
 collaboration identification, 164–165

V–Z

Variations, flexible design
 creating knobs for developers to turn, 337–338
 hot spots, recording on cards, 324–327
 hot spots, solving, 327–328
 inserting design placeholders, 335–337
 placing variable information into information
 holders, 334–335
 strategies for realizing variations, 329–330
 supporting with template and hook methods,
 330–333
 times needed, 333–334
Visitor design pattern, object collaborations, 171