Index

A
Abstraction filters, 105
Action-restricting rules, 157
Action-triggering rules, 157
Activity diagrams, 30, 31
Actor filters, 86
Actors
  definition, 37
  hierarchies, 86
  identifying, 72–73
  observers, 39–40
  real estate management example, 186–187
  refining, 97–98
  repetitions, 99–100
  role relationships, 38–40
  secondary, 39–40
  system interactions, 36. See also Facade iteration; Filled iteration.
Adaptability, 131
Adaptivity, 131
Aircraft carrier example, 142–143
Alternative paths, 44–45, 48, 49, 99, 144
Analysis, 4
Analysis models, traceability, 153
Application architecture, traceability, 154
Application traceable requirements, 156–157
Architectural framework, 156–157
Architecture traceable requirements, 156–157
Associations, 40–42
Assumption management, 113
Assumptions, use case template, 45
Author, use case template, 46
B
Baselines, 147
Black-box perspective, 25–26
Blogs, 151
Books and publications
  About Face: The Essentials of User Interface Design, 83
  Applying UML and Patterns, 28
  The Dream Society, 176
  Object-Oriented Software Engineering, 26
  Rapid Development, 12
  The Real Options Solution..., 148
  Requirements by Collaboration..., 17
  Software Architecture in Practice, 157
  The Unified Modeling Language, 28
  The Unified Modeling Language User Guide, 28
  The Unified Software Development Process, 127
  Zen and the Art of Motorcycle Maintenance, 35
Boundary and control classes, 153
Budgets, and the HI/I method, 137–138
Build-deliver-learn method, 130
Business process definitions, 71
Business rules
  adding, 101
  atomic nature of, 61
  categories of, 60
  Filled iteration, 101
  glossary of terms, 98
  traceability, 157
  use case template, 46
Business rules catalog
  Facade iteration, 81, 82
  sample, 60–62
Business use cases, 177–181

C
Calculations, traceability, 157
Change management, 112, 125
Chaordic states, 133
Class diagrams, 30
Classic mistakes
  context, 168
  messiness, 163–164
  mismanagement, 168
  notation, 170–171
  overengineering, 165–167
  perspective, 161
  thriftiness, 162
Classic view, 35
Class-responsibility-collaboration (CRC), 154
Coffee shop example, 121–123
Collaboration diagrams, 29–30, 153
Communicating via e-mail, 70
Complex adaptive systems, 132–133
Component object diagrams, 30, 32
Construction, 4
Context, classic mistakes, 168
Context-free use cases, 37
Contract-style requirements lists, 17–19
Customer/user identification, 71–72

D
Data owners, 66
Dataflow diagrams (DFDs), 23
Dates, HI/I method, 137–138
Deployment, 5
Deployment diagrams, 32, 33
Design
  definition, 4
  separating from requirements, 8–9, 60
Design models, traceability, 153
Developer goldplating, 15
Development, software. See software development.
Divide and conquer, 138–141, 143
Documentation and training,
  traceability, 155

E
E-mail communication, 70
Entity classes, 153
Entity-relationship diagrams (ERDs), 23–24
Estimating by doing, 147
Examples. See also real estate management example.
  aircraft carrier, 142–143
  coffee shop, 121–123
  instant messaging encryption, 225–228
  integrated systems, 219–223
  jet fighter, 142–143
  ordering from a catalog, 229–234
  requirements lists, 3–5, 6–8, 17–19
Exception paths, 100–101
Executive sponsor approval, 72–73
Executive sponsor’s viewpoint, 69–71
INDEX

F
Facade filters, 89
Failure rates, software development, 11
Focusing use cases, 110–112
Functional decomposition, 22
Functional requirements, 9, 59

G
Generalization, 41, 105
Granularity, 94–95. See also statement of work.

H
Hierarchy killer, xviii, 84–86
Hierarchies, 84–86
Holistic iterative/incremental (HI/I) method. See HI/I (holistic iterative/incremental).
Holistic method, 130–131

I
Implementation-specific language, 36–37
Includes, preconditions, assumptions (IPA) filters, 104
Incremental development, 129–130
Incremental methodology, 127
Industry experts, 65
Inferences, traceability, 157
Inquiry-only systems, 49–50
Internal customers, 72
Interviewing users, 15–16
Iterative development, 128–129
Iterative method, 127
Iterative/incremental lifecycles, 4–5. See also Facade iteration; Filled iteration; Focused iteration.

J
Jet fighter example, 142–143
Joint application design (JAD) sessions, 16–17
Joint requirements planning (JRP) sessions, 16–17

L
Language for modeling use cases. See OML; UML.
Language in use cases
implementation-specific terms, 36–37
noun filters, 88
readability, 175
refining, 101
verb filters, 87–88
Lifecycle activities (use case development), 4–5. See also Facade iteration; Filled iteration; Focused iteration.
Lifecycle methodology, 26

M
Maintenance, definition, 5
Merging duplicate processes, 110–111
Messiness, classic mistakes, 163–164
Mismanagement, classic mistakes, 168
Mission, vision, values (MVV), 56, 67
Mistakes, common. See classic mistakes.
Mock-ups. See prototypes.

N
Nonfunctional requirements.
defining, 76–77
definition, 9
documenting, 78, 80–81
Facade iteration, 74–81
identifying, 75, 78
ranking importance of, 78
template for, 79
traceability, 156–157
use cases for, 59
validating requirements, 78
vs. system maintenance requirements, 80
Non-object-oriented systems, 50
INDEX

Notation. See also OML; UML.
  classic mistakes, 170–171
  vs. methodology, 27
Noun filters, 88

O
Object diagrams, 30
Observers, 39–40
Opponents/supporters, identifying, 68
Other people’s views (OPV), 71
Overengineering, classic mistakes, 165–167

P
Packages, 32–33, 88
Paths. See also Basic Course of Events.
  alternative, 44–45, 49
  exception, 45, 49
  most common, 44
  in use case template, 44–45
Peer reviews
  Facade iteration, 89–90
  Filled iteration, 105
PERA. See Purdue Enterprise Reference Architecture.
Perspective, classic mistakes, 161
Portfolio management, 148
Postconditions, 46
Preconditions, 45, 96–97
Problem statement, 70
Product marketing, traceability, 155
Project history, reviewing, 67–69
Project management. See also HI/I;
  waterfall lifecycle.
  incremental methodology, 127
  iterative method, 127
  RAD (Rapid Application Development), 125–126
  Spiral lifecycle, 126
  staged delivery, 126
  traceability, 155
Project team, 64–65
Project visibility, 69
Proprietary names, 115–116
Prototypes
drawbacks, 19–20
HI/I (holistic iterative/incremental)  
  method, 144–145
  requirements gathering, 57, 59
  as requirements tools, 25
  user interface metaphors, 81, 83
  user interface storyboards, 83
Purdue Enterprise Reference Architecture (PERA), xvii, 130–131

R
Real estate management example
  actors, 186–187
  IRR (Internal Rate of Return),  
    calculating, 213–215
  overview, 183–184
  refining requirements, 212–218
  scope decisions, 187–189
  use cases, 184–186, 190–212
Real use cases, 37
Release planning, traceability, 155
Repetitions, 99–100
Requests for proposals (RFPs), 50
Requirements. See also scenarios; use cases.
  audit trails. See traceability.
  definition, 5
  description, 6–9
  Dr. Suh’s Minimal Good System
    Axiom, 85
  functional, 9, 59
  nonfunctional, 9, 59
  separating from design, 8–9, 60
Requirements definition, 10, 13
Requirements gathering
  common failings, 6
INDEX

definition, 4, 10
developer goldplating, 15
drawbacks, 1–5
eliminating redundancy, 14
ensuring traceability, 14–15, 59–60
guiding principles, 53–55
MVV (mission, vision, values), 56
premature assumptions, 12, 60
prototypes, 57, 59
reducing volume, 14
resolving conflicts, 12
risk analysis, 57, 58
scope definition, 175. See also granularity; statement of work.
statement of work, 57
steps involved, 55–56
use-case-driven, 53–55
user needs, 12–13
Requirements gathering, issues
JAD (joint application design) sessions, 16–17
JRP (joint requirements planning) sessions, 16–17
prototypes, 19–20
users, availability of, 12–13
users, interviewing, 15–16
Requirements lists
bad examples, 3–5, 6–8, 17–19
contract-style, 17–19
in use case development, 22
Requirements specification, 10
Return-on-investment (ROI), 148
Reviews
Facade iteration, 89–90
Filled iteration, 105
Focused iteration, 113–114
peer, 89–90, 105
of previous decisions, 68
project history, 67–69
user, 90, 105
Risk analysis, 57, 58, 81
Risk management, 113
Roles (actors)
naming, 40
and organization charts, 86
Roles (project team)
Facade iteration, 90–91
Filled iteration, 106
Focused iteration, 116
traceability, 14–15
Romantic view, 35
S
Scenarios
definition, 47–49
testing use cases, 101, 105
Scope definition, 175, 187–189. See also granularity; statement of work.
Secondary actors, 39–40
Security, traceability, 155
Sequence diagrams, 29–30, 153
Service use cases, 176–177
Sign-offs, 147
Software development
commonly ignored activities, 2
failure rates, 11
lifecycle activities, 4–5
phases. See lifecycle activities.
Software package evaluation, 50
Staged delivery, 126
Stakeholder interviews, 72–73, 104
State diagrams, 30
Statement of work, 57, 81. See also granularity.
Stereotyping, 34
Storyboards, 83
Storytelling with use cases, 175–176
Structural facts, traceability, 157
Subject matter experts (SMEs), 64
Summary, use case template, 43
Supporters/opponents, identifying, 68
Surplus functionality filters, 114, 115
System context level use cases, 85
System/actor interactions, 36. See also
   Facade iteration; Filled iteration.

T
Technology-centric vs. user-centric
solutions, 66
Test model, traceability, 154
Testing
   definition, 4
   filled use cases, 101, 105
Thriftiness, classic mistakes, 162
Traceability
   action-restricting rules, 157
   action-triggering rules, 157
   aids to, 150–151
   analysis models, 153
   application architecture, 154
   application traceable requirements,
      156–157
   architecture traceable requirements,
      156–157
   to business rules, 157
   calculations, 157
   CRC (class-responsibility-
collaboration), 154
   definition, 150
   design models, 153
   difficulties of, 150
   documentation and training, 155
   HII (holistic iterative/incremental)
      method, 144–145
   importance of, 150
   inferences, 157
   to nonfunctionals, 156–157
   potential problems, 150
   product marketing, 155
   project management, 155
   by project team role, 14–15
   release planning, 155
   role of use cases, 59–60
security, 155
structural facts, 157
test model, 154
tools for, 151
use cases, 175
to use cases, 152–155
user interface design, 154
Triggers, 45, 96

U
Usability, studies vs. designs, 83
Use case diagrams
   definition, 29
   example, 29
   Facade iteration, 84
   Filled iteration, 94
   showing relationships, 38–42
Use case granularity, 94–95
Use case names
   refining, 97
   use case template, 43
Use case surveys, 73–74
Use case templates
   alternative paths, 44–45
   assumptions, 45
   author, 46
   business rules, 46
   exception paths, 45
   iteration, 43
   layout, 46
   most common path, 44
   postconditions, 46
   preconditions, 45
   summary, 43
   triggers, 45
   use case names, 43
Use cases. See also requirements; scenarios;
   UML.
   adding details, 94–95
   advantages of, 174–176