Index

A

Abstract Window Toolkit, 110
Acceptance Plan artifact, Inception review, 245
Acceptance tests, 88, 199
Accessor classes, 159, 228–230
Achievement, recognizing, 27–28
Activities, RUP, 240–242
Actors, and use cases, 66, 67
Agile Software Development, 3, , 253–255
   Agile Alliance, 3
   Agile Manifesto, 3
Alpha release, 152
Analysis & Design discipline, artifacts, 242
Anonymous class in Java, 118
Architecture, 12, 83, 106
   executable, 84–85, 105–106
   PSP tools, 91–95, 111–113
   Software Architecture Document, 230
   stability of, 82–83
Artifacts, 241
   in Analysis & Design discipline, 242
   Construction phase and, 141, 247
   Elaboration phase and, 246
   in development case, 49–50
   Inception phase and, 245
   key, for every team, 11–13
   Transition phase and, 248
Assert class in JUnit framework, 129
Assertions, in unit tests, 129, 130
Assessments, and iterations, 17
Attribute matrix
   for feature requirements, 102
   for PSP Tools feature, 65
Automated testing, 114, 165
AWT. See Abstract Window Toolkit

B

Beck, Kent, 126, 254, 253–255
Beta testing, 141, 194, 195
Beta versions of tools, using on project, 110
BlueJ IDE, 234
Borland JBuilder, 234
Brochures
   PSP Tools, 62
   writing, 61
Brown thumb, 200
Bugs, fixing, 198
Business case, 57, 245
Butcher, Susan, 24

C

Cancel button, implementing, 178, 179
Capability Maturity Model, 4n2, 249
Carnegie Mellon University, Software Engineering Institute at, 249
Change control board (CCB), 198
Change management, 162–163
Change request management tool, 51
Class diagrams, 229, 230
Classes
   accessor, 159, 228
   anonymous, 118
   helper, 112
   testing, 126
ClearCase. See Rational ClearCase
Cleese, John, 218
Cloudscape, 99, 110
Cloudview, 99, 125, 151, 156
CMM. See Capability Maturity Model
CMU. See Carnegie Mellon University
Coaching, effective, 43–44
Code, 80
  accessor classes, 159, 228–230
  Construction phase, work during, 160
  getter methods, 179, 180
  helper classes, 112
  helper methods, 180
  I/O methods, 180
  menu construction, 170
  menu handling, 119, 173
  pair programming and, 163–164
  setter methods, 180
  smelling bad, 159
  Transition phase, changes during, 199
  user interface, 116–122
Code coverage, 78
Code structure. See Architecture
Coding standards, XP and, 255
Collaboration, team communication and, 231
Collective ownership, XP and, 253
Communication on team, 10, 21, 231
  fear factor and, 148
  during Transition phase, 197–198
  XP and, 253
Companion role, 44
Configuration management, 136
Configuration management tools, 77, 111
Configuration specification (ClearCase), 111
Construction phase, 105, 106, 135–167, 239, 243
  artifacts, 141, 247
  defect tracking and, 138
  development speed-up in, 157–160
  environment adjustments during, 135–136
  fine-tuning user interface during, 169–181
  finishing database during, 181–188
  goals of, 141, 194, 247
  implementation and, 145–149
  iterations, 152–163
  planning, 141–145
  programming in pairs and, 163–164
  requirements management and, 139–140
  testing during, 164–165, 189–192
  version control and, 136–137
Consultants, 45
Context menus, adding, 156–157, 174–176
Continuous integration, XP and, 254
Courage, XP and, 254
Coverage tests, 78, 165
Customers
  acceptance tests and, 88
  delighting, 152–154
  feedback from, 211
  focus on, 28, 36, 71, 101, 102, 104
  for PSP Tools, 7–8
  Inception phase and, 80–81
  inclusion of, 162
  on-site, XP and, 255
  post-mortem reviews and, 215
  supporting, 209–210
  training, 207–209
  XP and, 254–255
Database for PSP Tools
  accessor, for each data object class, 159
  alternatives to early implementation, 99–100
  classes, relationships, 230
  Construction phase and, 155, 181–188
  creation, deferring, 98–101
  design, 100–101
  Elaboration phase and, 110–111
  migration, 151
  package, class diagram for, 229
  schema, updating, 151, 181–187
  tables and fields diagram, 125
  version checking, during initialization, 184
Data loss defects, 160
Datapoint Technology Center (DATEC), 201
DefaultTreeModel in Swing framework, 122
Defects
  data loss, 160
  fixing, 160
  tracking, 52–53, 138
Delighters, 152–154, 162
Delivery dates, 194
Demands, ineffectiveness of, 26
Design Model, Elaboration and, 246
Design practices
  alternate data storage classes and, 99
  avoiding two-way dependencies, 113
Development Case, 12
  artifacts and, 49–50
  conventions used in, 46–47
  format for artifact table in, 46
  importance of, 50
  role map from, 47–49
  writing, 45–50
Development environment
  collaborative tools, 137
  configuration management and version control tools, 77, 136
  defect tracking, 138
  language tools, 75–76
  requirements management tools, 76–77, 139–140
  setup, 75–79
  testing tools, 77–78
  tools in Elaboration phase, 111
  tools in other environments, 78
Development processes, using on small projects, 9–18
Disabling or enabling menu items, 171–174
Disagreement, constructive, 24–25
Discipline for Software Engineering, A (Humphrey), 4n1, 58, 250
Distributed development teams
  challenges, 41–42
  instant messaging software for, 164
  tools for hosting projects for, 231
  using Groove to communicate, 51
Documentation, writing, 114–115, 204–207
Dollar-boxed releases, 249
Dynamic views (ClearCase), 137n2

E
Eclipse, 127, 234
  evaluating use of, 220
  IDE, 70, 95, 96, 117, 227
JUnit test results viewed in, 133
  running unit tests and, 132
Egos on development projects, 20
Elaboration phase, 13, 81–107, 169, 239, 243
  acceptance test plan for, 88, 89
  assessing, 104–106
  beginning, 109–113
  goals of, 81–87, 100, 245–246
  infrastructure development during, 158
  Iteration plan, excerpt from, 84
  iterations, 72, 73
  review viewlet, 106
  testing during, 164
  UI early in, 115
Enabling and disabling menu items, 171–174
Engineering backlog, managing, 97–98
Evaluating iterations, 17, 54
Evolutionary software development, 14
Exception–handling code, 226
Executable architecture
  producing, 84–85
  reviewing, 105–106
Exploratory testing, 90
eXtended Developer Experience. See Rational XDE
eXtended Markup Language. See XML
eXtreme Programming, 2, 3, 193, 195
  pair programming and, 163
Planning Game from, 82
  practices, 254–255
  test–first practice in, 87
  unit testing and, 126
  values, 253–254

F
Facilitators, for post–mortem reviews, 217–218
Fear factor, 146–148
Feature requirement attribute matrix,
  portion of, 102
Features, specifying as requirements, 63
Feedback
from PSP Tools customers, 211
from readers of this book, 235
XP and, 254

Fields
database schema changes, and, 181–187
handling single quotes in, 187

File menu, implementing, 120, 153, 170

Formality of process, 9–10, 15–16, 32, 45, 69

Forms, Personal Software Process, 251

Forte (for Java), 76, 116, 234
Community Edition, 111
switch from, to XDE, 95–97

40–hour week, XP and, 254

Freeze dates, 114

Garbage collection, 150

Geographical distribution, dealing with,
41–42, 51–52

Getter methods, 179, 180

Glossary artifact, and inception, 245

Goal setting, with Vision, 58–63

Gold owner, 61

Greenfield development, 157

Groove, 51–52, 78, 95, 97–98, 104, 105, 223
defect tracking with, 53, 138
Engineering Backlog and Defects tools in,
142–143
evaluating use of, 220, 221, 231
using for version control, 137

GUI builder, 76, 95–97, 117, 234

Guru role, 44

Hawthorne Effect, 202–203

Help, online, coding, 155, 170

Helper classes, 112

Hijacked files (ClearCase), 137

Humphrey, Watts, 249–250

IBM, acquisition of Rational Software, 11
IDE. See Integrated development
environment

Inception phase, 39, 57–80, 104, 239, 243
activities undertaken during, 57
code writing and, 79–80
development environment setup, 75–78
goals of, 80, 244–245
initial use cases defined, 66–71
iteration evaluation, 79
iteration plan, 53–54
iterations, 72
project management, 71–75
requirements management, 63–65
testing during, 164
use cases at end of, 67

Vision, 58–63

Independent testers, importance of, 199–203

Informal Iteration Plan template (RUP), 54

Information architecture, 6

Initialization, database version checks
during, 184

Initial Operational Capability (IOC)
milestone. See Construction phase

Instant messaging software, 164

Integrated development environment (IDE),
35, 70, 76
recommendations, 234
running unit tests and, 132
selection of, 111

IOC milestone (Initial Operational
Capability milestone). See

Construction phase

I/O methods, 180

Iteration Plan, for Inception phase, 53–54,
245

Iteration Plan template, RUP 2003, 54, 56

Iterations
assessments at end of, 17, 214
Construction, 51, 152–163
designing, 13
evaluating, 79
phases divided into, 239
planning and, 72
rhythms and, 142
RUP and description of, 14
Iterative development, 12, 13–15, 141–143

J
Java, 6, 120, 225, 234
Java Database Connectivity, 71, 98, 110
IDEs, 76, 234
platform options, 110
Swing, 110
Swing components, user interface structure and, 116
JDBC. See Java Database Connectivity
J2EE platform, working with, 99
JUnit, 78, 90, 127, 132, 164, 221
anatomy of JUnit test, 128–130
plug–in, 97
writing tests with, 128–132
test framework, 87
test results viewed in Eclipse, 133

L
Late changes, avoiding, 195–196
LCA milestone (Lifecycle Architecture milestone). See Elaboration phase
LCO milestone (Lifecycle Objectives milestone). See Inception phase
Learning environments, providing, 23
Learning periods, 157–158
Learning reflections, by team members, 221–225
Lifecycle Architecture (LCA) milestone. See Elaboration phase
Lifecycle Objectives (LCO) milestone. See Inception phase
Lines-of-code (LOC) counting, 251
Listeners, in code, 117–122, 179
Listening to team members, 43
LOC counting. See Lines–of–code counting
Login, implementing in PSP Tools, 94–95
Losing team members, 42–44
Loss of data defects, 160

M
Martin, Bob, 29, 149
MDD. See Model–Driven Development
Meetings
team formation and, 41
post-mortem, 213–235
Mentor role, 16–17, 44
Menu items
enabling and disabling, 171–174
implementing, 119, 152–154
naming and grouping, 170–171
Menus
adding items to, 120
context, 174–176
implementing, 170–174
pop–up, 174
Metaphor, XP and, 254
Microsoft NetMeeting, 105
Microsoft Project, 51, 54
Model-Driven Development, 68
Models of program architecture, 68–70,
227–230
Model-View-Controller pattern, 176n2
MyRUP, 232	no personal view created with, 233

N
Nonfunctional requirements, identifying, 70–71

O
Object–oriented database management system (OODBMS), 123n6
Ownership, requests and, 27

P
Package distribution, initial, 93
Packages, in PSP Tools, 112
Packaging a product, 195, 203–207, 211
Pair programming, 43n5, 163–164, 254
Patterns in code, 160
People, importance of to success of project, 19–28, 37–38
Personal Software Process (PSP), 1, 2, 5, 51, 58, 249–252
components, 249–251
goals, 251–252
individual developer, and, 251
levels, 250, 251
process structure, 252
purpose, 101
Planning, 71–74
during Construction phase, 142
during Inception phase, 54–55
Planning Game, XP, 253
Pop-up menus, 156–157, 174–176
Postmortem review, 213–235
action items produced from, 218–219
agendas for, 215–216
assessing PSP Tools project, 220–225
benefits of, 213–214
changes resulting from, 231–234
conducting, 214–219
facilitators for, 217–218
format for action items from, 219
fun activities after, 219
goals for, 215–216
preparation guidelines for, 216–217
post-project activities, 225–230
sample agenda, 216
Precision requests, 26, 41n3
PRM. See Product Release Milestone
Problem statement, from Vision, 59
Process, 19, 28–34, 37–38
configuring, 31, 32–33
customizing, 36
prime directive of, 28–29
reuse, 30
risk-driven, 29
starting points, 30–31
tailoring, 31–33
Process Views, RUP, 232
Productivity, well-formed requests and, 27
Product Position statement, from Vision, 60
Product Release Milestone, 247
Programmers, learning environments for, 23
Programming language tools, selecting, 75–76
Programming, in pairs, 43n5, 163–164
Progress reporting, 51–53
Project plan, 13, 50, 72, 73
Projects
balancing resources, 36–37
deciding scope of, 58–60
key artifacts, 11–13
leaders, team involvement and, 16
management, 71–75
members, forming a team, 39–56
phases, team evolution through, 166
Project Specific Templates and Tools,
Elaboration review and, 246
Prototypes, Elaboration review and, 246
PSP. See Personal Software Process
PSP Tools, 8, 40, 106, 198, 221, 231
Alpha release, 152
architecture, 91–95
building, 135–167
code organization, 111–113
database structure, 123–125
database updates and, 181–187
data model, 100
displaying time and defect data, 176–179
at end of Elaboration phase, 146–150
features, attribute matrix for, 65
future of, 234–235
infrastructure code, 158–160
installation of, 103–104
Java platform choices and, 110
login and, 94, 95
main window, key Swing items in, 116
menus, setting up, 119
test suite, 130–131
User’s Guide, 205–206
user interface, 91–93, 113–122
user training and, 207, 208
PureCoverage. See Rational PureCoverage
Q
QA team, collaboration with documentation team, 114–115
Quality, 19, 87, 164, 200

R
Rational ClearCase, 77, 111, 136
configuration specification, 111
dynamic views, 137n2
hijacked files, 137
snapshot views, 137
VOBs, 111, 136
Rational ClearQuest, 51
Rational Developer Network, 18n6
Rational PureCoverage, 78, 165
Rational RequisitePro, 63, 64, 77, 102, 139, 162
attribute matrix for features, 65
traceability matrix from, 140
Vision document excerpt in, 64
Rational Rose, 70, 86
Rational Unified Process, 237–248
activities, 240–242
benefits of, on small projects, 10–11
Development Case in, 45
getting started with, 11–16, 17–18
hump chart, 135, 195
Iteration Plan template, 54
iterations described by, 14
key artifacts and, 11–13
level of formality with, 15–16
milestones, 243–248
new features in RUP 2003, 232–233
process structure, 237–239
Process Views, 232
quality and, 87, 164
roles, 239–240
Test Plan Template from, 88
unit testing and, 126
Rational XDE, 70, 86, 95, 124, 220, 227
switch from Forte to, 95, 98–99
Rational XDE Tester, recording tests with, 159
Refactoring code, 158, 225–227
Release dates, determining, 203–204
Release notes, 143–144, 249
Requests
deleing, 40
power of, 25–27
well-formed, 40, 41
Requirements
dchanges, during Transition phase, 197
defining, 201
details added to, 85–87
formality of, 15
nonfunctional, 70–71
prioritizing, 71–75
stakeholders, involving, 60–61
testing and, 12
use cases and, 12, 66–68
Vision and, 58–63
Requirements management, 63–65
Construction phase and, 139
tools, 63, 76–77
Transition phase and, 197
Requirements workflow, 243
RequisitePro. See Rational RequisitePro
Respect, importance of on team, 21
Reverse-engineering code, 227, 228
Rhythm of PSP Tools project, 141–143
Risk List artifact, Inception review and, 245
Risks, managing, 11–12, 74, 94
addressing in Elaboration phase, 82–85, 98–100, 106
architecture and, 82–83
identifying in Inception phase, 74–75
prioritizing use cases based on, 84–85
process and, 29
sample, 75
Roles, 239–240
role map, 47–49
RUP. See Rational Unified Process
RUP Organizer, 234n2
SAD. See Software Architecture Document
Scenarios, 106, 164, 165
Schedules of projects, 13, 17
Scope and objectives section, in Iteration Plan, 54
Scope management, 101–103
Screen shots, minimizing use of in documentation, 114
SCRUM, 2, 3, 82, 97
SEI. See Software Engineering Institute
Sequence diagrams, 70, 85–87
Setter methods, 180
Shortcut menus, 156–157, 174–176
Simplicity, XP and, 253
Single quotes, handling in database fields, 187
Small projects
development process used on, 9–18
key characteristics of, 9–10
RUP and benefits for, 10–11
Snapshot views (ClearCase), 137
Software acceptance tests, 87
Software Architect, 221, 240, 242, 244
Software architecture, See Architecture
Software Architecture Document, 230, 246
Software development
communication and, 21
lifecycles, 13
people and, 19
risks and, 74–75
spiral model of, 14
teams, composition of, 21–22
Software Development Plan, 72n6, 245
Software Engineering Institute, 4, 249
SourceForge, 137, 231
Spiral model, of software development, 14
Spirit of RUP, 11
SQL, 98, 123, 187
SQL utility methods, using, 188
Srinivasan, Raj, xiv, 165, 189, 200, 203
Stability of requirements and architecture, 82–83
Stakeholders
acceptance testing and, 141
creating value for, 28, 45
creating Vision with, 59-61
determining what to build for, 57
Gold owner, 61
identifying, 60-61
Inception phase and, 80, 244–245
influence on UI, 156
part of team, 148
post-mortem review, and, 215
Transition phase and, 195, 197
verifying architecture with, 106
Storyboards, 80
Structured Query Language, See SQL
Sun ONE Studio, 76, 234
Supplementary Specifications, Elaboration review and, 246
Supporting customers, 209–210
Swing (Java), 110
T
Tables, creating in Java, 176-179
Teams
advisory roles to, 44–45
changes in membership, 42-44, 165-166
composition, 21–22
constructive disagreement in, 24–25
forming, 16-17, 39–44, 56
listening to, 43
trust within, 23–24
well-formed requests in, 25–27
Team Software Process, 251
Technical writers, role on project, 205, 207
Templates
Iteration Plan, 54
RUP, 46
RUP 2003 Iteration Plan, 56
Test Plan, 88
Vision, 58
Test automation, 165
Test discipline, in Rational Unified Process, 164
Test-driven development, 91, 126
Testers, independent, 199–203
Test–First Programming, 45, 87, 91, 126, 202, 221, 224, 227
Testing, 12, 109, 211
acceptance, 88, 199
during Construction phase, 164–165, 189–192
coverage, 165
independent testers, value of, 199–203
one’s own software, 200–203
test plan creation, 87–91
tools, 77–78, 114
unit, 126–133
user interface changes and, 114
using Rational XDE Tester, 186
XP and, 253
Test Plan Template, 88
Test report sample, 192
Test suites, using to group tests, 130–132
TFP. See Test–First Programming
Thin plug–ins, with RUP 2003, 233
Time–boxed iterations, 142n6, 239
Time–boxed releases, 249
Tools, 19, 34–37
building from scratch, 35–36
configuration management, 77
environment, changes to, 95–98
JUnit framework and, 128–130
language, 75–76
learning how to use, 35
in other environments, 78
people and process supported by, 34
requirements management, 63, 76–77
unit testing, 127–132
version control, 77–78
Traceability, 139–140
Training users, 195, 207–209
Transition phase, 94, 141n5, 193–211, 239, 243
code changes during, 199
customer support and, 209–210
description of, 193–194
future releases and, 210
goals of, 195, 247–248
making transition to, 194
packaging product and, 203–207
requirements and, 195–198
short, 196–197
test automation and, 165
training users and, 207–209
Trees, creating, 120–122
tree items, displaying pop–up menu for, 175–176
TreeModel interface, 122
Trust, importance of on team, 21, 23–24, 45, 149
TSP. See Team Software Process

U
UML. See Unified Modeling Language
Unified Modeling Language, 11, 123
diagrams, 97, 227–230
object–oriented systems, and, 123
Unit testing, 90–91, 126–133, 164, 224, 227
assertions used in, 129, 130
automating, 127
framework, 127
running, 132–133
Usability, enhancing in products, 151–152
Use–Case Model artifact, Inception review and, 245
Use cases, 12, 80, 106
details in, 85
at end of Inception phase, 67
defining for PSP tools, 66–69
Use Cases,
for Open Project Database, 85, 86, 98
in Iteration Plan, 54
prioritizing, 72, 85
testing, and, 189
and user documentation, 205
User documentation, 204–207
User interface
aesthetics, 156
building with Java, 110
code for, 116–122
User interface, continued
  design, 6
  effect of programming changes on, 113–115
  fine-tuning, 169–181
  initial sketch of, 92
User preferences, using, 179–181
Users, training, 195, 207–209
User’s Guide, 205, 207, 211
  table of contents, 206

V
Verification points in automated testing, 189
Version control, 111
  Construction phase and, 136–137
  tools, 77
Versions, planning for future versions, 210
Viewlets, 105, 106
Views (ClearCase), 111, 137n2
Vision, 11, 57
  brochure and, 61
  discovering extent of project, 58–60
  excerpt, in RequisitePro, 64
  features specification and, 63
  Inception phase and, 58–63, 245
  problem statement from, 59
  Product Position statement from, 60
  releases and, 102, 103
  stakeholders and, 60–61
  VOBs (ClearCase), 111, 136

W
Waterfall lifecycle, 13–14
  risk profile for iterative lifecycle versus, 15
Web–based training modules, 209
WebEx, 105, 209
WebSphere (IBM), 234
Well–formed requests, 26, 40, 41
Workflow details, 241
Workshops, scheduling for teams, 16–17
Writing Development Case, 45–50

X
XDE. See Rational XDE
XML, user preferences files and, 179, 181
XP. See eXtreme Programming

Y
YAGNI (You Aren't Gonna Need It), 94