

INDEX

A

Abstractions, key, 324

Acceptance tests. *See* Product acceptance tests

Activities

architect, 323–328

defined, 13, 383

iteration, 238–239

overview of, 14–15

project manager, 280–282

RUP lifecycle, 88–90

SEI CMM framework, 55

steps in, 15

tester, 373–379

Actors

defined, 383

detailing in Inception, 101–102, 299

identifying/describing in Inception, 98–100,
297–298

Adaptive development, 51–52

Adopting RUP, 197–222

assessment, 199–200

configuring/customizing, 203–204

evaluating, 206–207

executing, 205–206

in large organizations, 208–215

overview of, 197–199

paying lip service to RUP vs., 252

planning, 200–203

in programs for major change, 217–221

in programs for moderate change, 215–217

tool automation and, 208

Adopting RUP, mistakes

customization, 251

doing too much, 244–246

not adopting incrementally, 246–249

not planning implementation, 249–250

overview of, 243–244

paying lip service, 251–252

process improvement and business results,
250–251

Adoption of the PMI Guide to PMBOK, 277

Agile Development Movement, 50–53

Analysis-paralysis, 264

Analysts, 287–310

business operations and, 289–292

fine-tuning and, 304–308

mission of, 287–289

requirement testing, 309

requirement updating/refinement, 308

resources for, 310

role of, 309

stakeholder needs and, 292–293

starting point for, 289

use-case model/glossary development,
296–302

use-case specification example, 302–303

Vision development, 293–296

“Application proxy” code, 122

Architect, 311–332

activities of, 323–328

architecture and, 314–319

developer working closely with, 334, 337

function of, 320–322

mission of, 311–314

resources for, 329–331

roles of, 319–320, 328

Architectural baseline

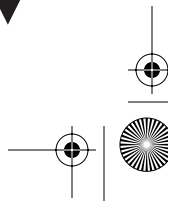
defined, 383

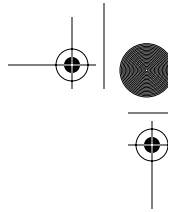
developing early on in project, 5, 38–40

by end of Elaboration, 266–269

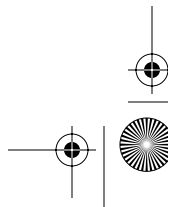
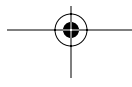
overview of, 122

Architectural coverage, 128–129





- Architectural integrity
 - architect's role in maintaining, 327–328
 - ensuring in Construction, 140
 - team communications and, 314
- Architectural mechanisms (patterns)
 - defined, 38
 - enforcing in Construction, 149
 - identifying in Elaboration, 130–131
 - overview of, 318
 - role of developer, 337, 355–356
- Architecture, 314–319
 - adopting RUP, 247–249, 252
 - component-based vs. functional decomposition, 40–41
 - costs of changes to, 37
 - defining, 38, 315
 - enforcing in Construction, 148–149
 - identifying key system functionality, 103, 105–106
 - iterative approach and, 8
 - layered, 121–122
 - models/views and, 315–317
 - Project Deimos and, 74
 - prototypes, 318
 - reusing, 266
 - reviewing, 327–328
 - risk and, 30
 - Software Architecture Document and, 317–318
 - teams organized around, 44–45
 - test automation, 373
 - testing and, 46–47
- Architecture, in Elaboration phase, 120–134
 - architectural coverage, 128–129
 - architectural mechanisms, 130–131
 - baselining, 266–269
 - consolidating and packaging classes, 127–128
 - critical use cases and, 123–127
 - database design, 129–130
 - describing run-time architecture, 130
 - implementation, 131
 - integration, 131–132
 - iterations in, 116–118
 - overview of, 120–122
 - subsystems/key components and, 122–123
 - testing, 132–133
 - user interactions in, 133–134
- Artifacts
 - adopting RUP and, 244–249
 - defined, 13, 383
 - Development cases specifying, 184–186
 - key test, 371–373
 - overview of, 15
 - producing executable software and, 34
 - product acceptance testing and, 174–175
 - RUP lifecycle phases and, 90–91
 - SEI CMM framework and, 55
- Assessment
 - adopting RUP and, 198–200, 207, 214
 - focus of tester in, 359
 - focus on inspections vs., 269–270
 - high ceremony approaches to, 54–57
 - iterative approach and, 9
- Automation, testing
 - focus of, 269–270
 - role of developer, 348
 - RUP philosophy, 368
- Automation, tool
 - addressing core problems, 208–209
 - high ceremony approach and, 60–61
- B**
- Bach, James, 361
- Base. *See* RUP base
- Baselining. *See* Architectural baseline
- Beck, Kent, 341, 355, 358
- Best practices. *See also* Guidelines
 - concept of, 52
 - defined, 19
 - developer, 354–357
- Beta release
 - preparing for deployment, 142, 157
 - Project Deimos, 81
 - testing, 162, 167–169, 174
- Bids, project, 255–256
- Bill of Materials (BOM), 172
- Boehm, Barry, 60, 240, 322
- Booch, Grady, 258, 279, 326, 358, xxxv
- Boundary classes, 340
- Brainstorming sessions, 98–100
- Brooks, Frederick, 314
- Builder. *See* RUP Builder



- Builds
 - adopting RUP and, 246–247, 249
 - defined, 225, 383
 - integration planning of, 352–353
 - iterative development and, 350
 - releases vs., 351
 - role of developer, 353
 - role of tester, 376–377
- Business Case
 - altering, 91
 - costs, schedule and risks in, 107–108
 - focus of, 250–251
 - motivating RUP, 199–200
 - multiple iterations and, 95
 - Project Deimos, 70, 73–74, 76–77
 - role of project manager in, 281
 - validating, 206
- Business modeling, 288, 289–292
- Buy-in
 - adopting RUP and, 250–251
 - business results and, 250
 - detailing requirements, 265
- C**
- Capability Maturity Model (SEI CMM), 54–56
- CCBs (Change Control Boards)
 - managing costs, 37
 - role of tester and, 379
 - supporting project manager, 276
- CCM (Configuration and Change Management)
 - adopting RUP incrementally, 247
 - high ceremony and, 61
 - managing costs, 37
- Ceremony, defined, 50
- Change Control Boards. *See* CCBs (Change Control Boards)
- Change requests
 - for defects and beta tests, 167
 - focusing on change management, 247
 - Project Deimos, 82
- Changes
 - accommodating early in project, 5
 - adopting RUP and, 197
 - agile development view of, 51
 - avoiding many late, 260–261
 - benefits of iterative development, 8
 - beta testing and, 167–168
 - guidelines for, 35–38
 - implementing major, 217–221
 - implementing moderate, 215–217
 - indicating completion of Elaboration, 268–269
 - Structural Plug-Ins creating, 194–195
- Classes. *See also* Design, use cases/components
 - architectural analysis, 324
 - consolidating, packaging identified, 127–128
 - implementing critical scenarios, 131
- Client/server architecture, 105–107
- CMM (Capability Maturity Model), 54–56
- CMMI, 55–56
- CMP (Core Message Platform), 173
- CMS (Configuration Management System)
 - overview of, 147–148
 - Transition beta testing using, 167–168
 - workspaces, 352
- Coarse-grained project plans, 226–228
- Code
 - continuous testing of, 150
 - implementing/unit-testing, 153–154
 - principles of approach to, 5
 - reviews, 347–348
- Collaboration diagram, 343
- Commercial Off-the-Shelf (COTS) components, 265
- Commitments, defined, 224
- Communication
 - organized around architecture, 145–147
 - role of architect, 312–314
 - role of project manager, 275
- Community/marketplace, 20
- Components. *See also* Design, use cases/components; Process Components
 - costs of changes to, 37
 - defined, 383
 - defining in Elaboration, 122–123
 - emulating with test stubs, 348
 - evolution of, 154
 - guidelines, 40–43
 - incrementally improving, 249
 - integrating in Elaboration, 131–132
 - principles of approach to, 5
 - reusing third-party, 266
 - role of developer, 347

- Components, Construction
 - completing design, 152–153
 - integrating in, 154–155
 - iterations in, 143–144
 - testing in, 153–154
- Concepts
 - Agile Development Movement, 52
 - defined, 16
 - key, 224–226
- Conceptual prototype, 305
- Concurrency
 - defined, 373
 - outlining in Elaboration, 130
- Configuration and change management. *See* CCM (Configuration and Change Management)
- Configuration Management System (CMS), 147–148, 167–168
- Configuration manager, 333
- Configuration (RUP Process Configuration)
 - adopting RUP, 198, 204, 207, 247
 - customizing templates, 184
 - defined, 385
 - deployment, 215
 - Development case and, 185–186
 - linking from project Web site to, 188
 - process map and, 58, 61
 - producing, 180–182
 - producing Process Views, 182–183
 - tools, 20–22, 80
- Constraints
 - adding to Vision statement, 79–80
 - stakeholder expectations and, 255
- Construction phase, 139–160
 - architectural enforcement, 148–150
 - artifacts, 90–91
 - beta deployment preparations, 157
 - Configuration Management System, 147–148
 - costs of changes at, 36
 - defined, 12–13, 384
 - deployments, 155–156, 157–159
 - designing database, 153
 - filling in design, 152–153
 - implementing/unit-testing all code, 153–154
 - Initial Operational Capability Milestone, 159
 - integration/system testing, 154–155
 - iterations in, 142–145, 232–234, 237–238
 - misconceptions about, 87–88
 - multiple bids and, 255–256
 - objectives/milestones, 11, 141–142
 - organizing around architecture, 145–147
 - outline of, 206
 - overview of, 139–141
 - process and tool enhancement projects at, 211–212
 - Project Deimos, 72
 - resdesigning database in, 350
 - risk mitigation in, 88–89
 - role of architect, 320
 - role of developer, 338
 - role of tester, 375
 - stabilizing architecture before, 266–269
 - use cases/requirements and, 151–152
 - workflows in, 89–90
- Contractual issues, 277
- Control class, 340
- Core Message Platform (CMP), 173
- Cost
 - estimating in Elaboration, 134–135
 - of introducing changes, 36–38
 - objectives of Construction, 142, 145
 - quality and, 362–363
 - regression tests minimizing, 155
 - RUP not dealing with issues of, 277
 - software testing and, 366
 - understanding in Inception, 107–108
- COTS (Commercial Off-the-Shelf) components, 265
- Crystal, 51–52
- Cunningham, Ward, 341
- Customizing, 189–196. *See also* RUP product
 - adopting RUP, 198, 203–204, 207
 - deployments, 215
 - iterative approach to, 251
 - overview of, 189–190
 - Process Engineering Toolkit and, 190–191
 - Rational Process Workbench and, 190–191
 - RUP templates, 184
 - structural RUP plug-ins, 192–196
 - thin RUP plug-ins, 191–192
- Cycle
 - defined, 384
 - overview of, 224

D

- Database Administrator (DBA), 333
- Databases
 - deploying/ converting in Transition, 172
 - deploying in Construction, 157–158
 - designing in Construction, 153
 - designing in Elaboration, 129–130
 - implementing/ testing by developer, 349–350
- DBA (Database Administrator), 333
- Defects
 - defined, 373
 - iterative approach to, 8
 - metrics for analyzing Transition, 169–170
- Delivery tools. *See* Process delivery tools
- Department of Defense (DOD), 56
- Deployment
 - Construction, 155–159
 - defined, 384
 - Transition, 172
 - views, 317
- Design
 - adopting RUP incrementally, 247
 - avoid stating in requirements, 265
 - classes, 131
 - costs of changes to, 37
 - coupling testing with, 47
 - of database in Construction, 153
 - of database in Elaboration, 129–130
 - emphasized in Construction, 141, 152–153
 - executable architecture, 120–121
 - implementing/ testing database, 349–350
 - mission of analyst, 288
 - role of architect, 325, 327
 - role of developer, 336–337, 356
- Design model, 345–347
- Design, use cases/ components, 338–346
 - analysis classes/ analysis model, 343–344
 - design classes/ design model, 345–347
 - distributing behavior to analysis classes, 341–343
 - first-draft outline, 339–341
 - overview of, 338–339
 - use-case design, 344–345
- Developers, 333–358
 - architect communications with, 314
 - best practices, 354–357
 - designing, implementing, testing databases, 349–350
 - frequent integration and, 350–353
 - limiting at project start, 255–258
 - mission of, 333–335
 - requirements/ design constraints, 336–337
 - resources for, 358
 - task overview, 335–336
 - testing, 347–349
 - use case/ component design, 338–346
 - use case/ component implementation, 347
- Development case
 - accessing on project Web site, 187–189
 - alternatives to, 189
 - deciding on process, 109
 - defined, 384
 - producing, 184–187
 - project manager's approach to, 279
 - refining in Elaboration phase, 115, 136–138
- Development cycle
 - overview of, 165–167, 224
 - possible degrees of iteration, 233
 - Transition and, 165–167
- Dikel, David, 320–322
- Disciplines
 - defined, 384
 - as high-level workflow, 16
 - list of, 17–18
- Distribution
 - defined, 373
 - physical, 130
- Documentation
 - evaluating RUP project, 207
 - functionally oriented teams and, 253–254
 - RUP test philosophy and, 368
 - stakeholder requests, 293–296
 - writing Vision, 294
- Documentation, Transition phase
 - improving, 168
 - packaged product, 172–173
 - product acceptance testing and, 174–175
 - training users, 171
- DOD (Department of Defense), 56
- DOD-STD, 56–57
- Domain models
 - defined, 296
 - understanding key information through, 299
 - updating towards end of Inception, 304



Dynamic Process Structure
defined, 9–10
overview of, 10–13

E

Elaboration phase, 113–138. *See also* Architecture,
in Elaboration phase
artifacts in, 90–91
breaking project into multiple bids, 255–256
costs of changes at, 36
defined, 12, 384
detailing actors/use cases in, 300–302
detailing requirements, 118–120
guidelines for testing, 46
iterations in, 116–118, 232–234, 237
misconceptions about, 87–88
objectives/milestones, 11, 114–115
outline of, 205–206
overview of, 113–114
process and tool enhancement projects at, 211
Project Deimos, 71–72
refining Development case, 136–138
risk in, 30, 88–89, 134–135
role of architect, 319–320
role of developer, 334, 338
role of tester, 375
verifying nonfunctional requirements in, 307–
308
workflows, 89–90

Encapsulation, 42

Entity classes, 340

Errors. *See* Mistakes

Estimations, project planning
overview of, 239–240
role of project manager, 280
Wideband Modified Delphi and, 240–241

Evaluation
adopting RUP and, 199, 207
deploying RUP and, 215
iteration planning and, 236
mission, 369
role of tester in, 377
test evaluation summary, 371

Evolution cycles
defined, 166, 224
iteration planning in, 237
staffing profiles in, 235–236

Evolutionary prototypes, 120–121

Executable architecture
incrementally improving, 249
overview of, 120–121

Executable software
incrementally improving, 249
paying lip service to RUP and, 252
staying focused on, 33–35, 269–270

Execution
adopting RUP and, 198–199, 207
deploying RUP, 215

Experience, architectural, 312

Extended Help
defined, 384
overview of, 24

Extreme Programming (XP)
as agile process, 51–52
RUP architecture compared with, 268

F

Feature list, 294

Feedback
continuous process improvement and, 221
early deployment in Construction and, 155–156
Transition beta testing and, 167–168

Fine-grained project plans, 226–228

Flaws. *See* Defects

Flow of events, structuring, 301–302

Functional decomposition architecture, 40–41

Functional organizational structure, 43

Functional requirements, 32–33, 336

Functionality, key system, 102–104

G

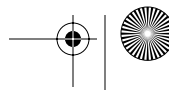
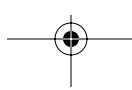
Generalizations, defined, 263

Generations, software, 165

GEQ (Good Enough Quality) concept
conforming to standards and, 364–365
cost and, 363
overview of, 361
paradigms of, 361–362
quantification and, 363–364

Gilb, Tom, 28, 285

Glossary
analyst developing, 296
creating in Inception, 100, 304
for this book, 383–387



- updating in Elaboration, 119
- Goals
 - adopting RUP incrementally, 246–249
 - architect, 312
 - ensuring progress in Construction, 150–151
 - time-boxing, 225–226
- Gold plating, 261
- Good Enough Quality. *See* GEQ (Good Enough Quality) concept
- Green-field development
 - defined, 224
 - iteration planning and, 237
 - staffing profiles in, 235–236
- Guidelines, 27–48
 - accommodating change, 35–38
 - adding, 18
 - addressing risks, 28–31
 - architect developing, 318–319, 327
 - components, 40–43
 - defined, 16
 - delivering value, 31–33
 - executable architecture, 38–40
 - executable software, 33–35
 - overview of, 27–28
 - programming language, 347
 - quality, 46–47
 - team work, 43–45
- H**
- High Ceremony approach
 - factors driving towards, 60–61
 - RUP using, 58–59
 - Waterfall/Iterative compared to, 50–51
- Human resources, 277
- I**
- IBM, 122
- IDEs (Integrated Development Environments), 265
- Implementation. *See also* adoption
 - avoiding pitfalls of, 249–250
 - costs of changes to, 37
 - objectives of Construction, 141
 - role of architect, 326
 - role of developer, 335, 347
 - unit-testing in Construction, 153–154
 - views, 316
- Inception phase
 - artifacts in, 90–91
 - avoid over-detailing requirements at, 264
 - breaking project into multiple bids, 255–256
 - costs of changes at, 36
 - costs, schedule, risks, 88–89, 107–108
 - decisions on process, tools, 108–110
 - decisions on what to build, 96–102
 - defined, 12, 384
 - determining architecture, 105–107
 - developing project rhythm, 258–259
 - identifying system functionality, 102–104
 - iteration planning, 95–96, 232–233, 237
 - Lifecycle Objective Milestone and, 110–111
 - misconceptions about, 87–88
 - objectives/milestones, 11, 94–95, 110–111
 - outline of, 205
 - overview of, 93–94
 - process and tool enhancement projects at, 211
 - Project Deimos, 71
 - risks, 88–89, 107–108
 - role of analyst, 297–299
 - role of architect, 319
 - role of tester, 375
 - testing guidelines for, 46
 - workflows in, 89–90
- “Includes”, defined, 263
- Industry sectors, 24–25
- Initial development cycles
 - iteration planning, 237
 - overview of, 224–225
 - staffing profiles in, 235–236
- Initial Operational Capability (IOC) Milestone, 11
 - overview of, 159
 - project plan including, 227
- Inspections, 269–270
- Instantiation, 184–189
 - alternatives to Development cases, 189
 - Development cases and, 184–187
 - project Web sites and, 187–189
- Integration
 - in Elaboration, 131–132
 - iterative approach to, 7
 - planning in Construction, 148–149
 - progressing in Construction, 150

Integration (*continued*)

- role of developer in planning, 350–353
- testing in Construction, 154–155
- workspaces, 352

Integration Build Plan, 352–353

Integration workspaces, 247, 352

Interface, component

- defined, 42
- defining in Elaboration, 122–123
- sketching/completing Project Deimos, 77–78

IOC (Initial Operational Capability) Milestone, 11, 159, 227

ISO/IEC 15504, 55

Iteration plans. *See* Iterative development, mistakes; Project plans

Iterations

- addressing top risks, 28–31
- avoiding overlapped, 259–260
- Construction and, 142–145
- defined, 225, 384
- Elaboration and, 116–118
- improving process/tools throughout, 205–206
- improving test assets after, 378–379
- Inception and, 95–96
- length, 234–235, 258–259
- number of, 232–234
- role of project manager, 279, 281–283
- RUP lifecycle phases and, 11–12
- team spirit and, 6
- test philosophy of, 368
- Transition and, 163–165

Iterative development

- focusing on, 246–247
- misconceptions about, 87–88
- RUP and, 6–9

Iterative development, mistakes, 253–261

- extending initial iterations, 258–259
- late changes, 260–261
- misleading stakeholders, 254–255
- number of starting developers, 255–258
- overlapping iterations, 259–260
- overview of, 253
- solving easy tasks first, 257–258
- using functional, specialized organizations, 253–254

J

- J2EE platform, 42, 62
- Jacobson, Ivar, 310, 330

K

- Key abstractions, 324

L

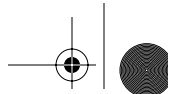
- Launch-packaging, 172–173
- Layered architecture, 121–122
- LCA (Lifecycle Architecture) Milestone, 11
 - overview of, 137
 - project plan including, 227
 - role of architect in, 319–320
- LCO (Lifecycle Objective) Milestone, 11
 - overview of, 110–111
 - project plan including, 227
- Leadership, architects and, 312
- Legal issues, 277
- Libraries
 - RUP Library, 182
 - RUP Process Library, 191
- Lifecycle Architecture (LCA) Milestone, 11
- Lifecycle Architecture Milestone. *See* LCA (Lifecycle Architecture) Milestone
- Lifecycle Objective (LOC) Milestone, 11
- Lifecycle Objective Milestone. *See* LCO (Lifecycle Objective) Milestone
- Lifecycle phases, 11–13
- Load testing, 132
- Logical views, 316
- Low Ceremony approach
 - Agile Development Movement and, 50–53
 - factors driving towards, 60–61
 - RUP using, 58–59
 - Waterfall/Iterative compared to, 50–51

M

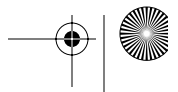
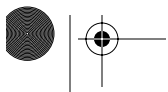
- Maintainability, defined, 373
- Maintenance cycles
 - defined, 224
 - staffing profiles in, 235–236
- Management. *See also* Project management
 - failing to set expectations for, 254–256
 - getting buy-in from, 249
- Managing iterative development, 8

- Marketing rollout, 173
 - Marketplace/community, 20
 - Matrix organizational structure, 43
 - Metrics
 - completing Transition and, 169–171
 - role of project manager, 280, 282
 - Microsoft.NET, 30, 42
 - MIL-STD, 56–57
 - Milestones
 - costs of changes at, 36
 - defined, 384
 - project plan including, 227, 230–231
 - role of project manager in assessing, 282
 - RUP lifecycle phases and, 11–13, 88–89
 - Military standards of software development, 57
 - Mission
 - of analysts, 287–289
 - of architects, 311–314
 - of developers, 333–335
 - of project managers, 273–276
 - statements, 369
 - of testers, 359–365, 375–376, 378
 - Mistakes
 - adopting RUP. *See* Adopting RUP, mistakes
 - analysis-paralysis, 264
 - ending Elaboration without stable architecture, 267–269
 - including design with requirements, 264–265
 - iterative development. *See* Iterative development, mistakes
 - not assessing executable software, 269–270
 - not reusing solutions/components, 265–267
 - stakeholder buy-ins, 265
 - too many use cases, 262–263
 - Modeler. *See* RUP Modeler
 - Monitoring, 282
 - Motivation, 223–224
 - MyRUP
 - defined, 384
 - overview of, 22–23
 - Process Views in, 182–183
- N**
- .NET platform, 30, 42
 - Non-functional requirements, 307–308
 - “Not invented here” mentality
 - developer best practices, 355–356
 - overview of, 265–267
- O**
- Objectives
 - Construction, 141–142
 - Elaboration, 114–115
 - Inception, 94–95
 - iteration, 235
 - motivating RUP implementation, 197, 199
 - RUP lifecycle phases, 90
 - RUP lifecycle phases and, 11–13
 - Transition, 162–163
 - Optimizations, 241–242
 - Organizations
 - adopting RUP in large, 208–215, 276
 - avoiding functionally oriented teams, 253–254
 - getting buy-in from management, 249
 - Organizer. *See* RUP Organizer
- P**
- Packaging, 172–173
 - Pair programming, 52, 356–357
 - Parallel development, 140
 - Partnering, 322
 - Patch releases, 169
 - Performance
 - evaluating, 199
 - improving future, 175
 - runtime analysis and, 348–349
 - RUP process and, 206–207
 - testing in Elaboration, 132
 - Persistency, 350
 - Personnel. *See* Team
 - Phases
 - defining, 224–225, 385
 - overview of, 87–92
 - role of analyst in, 289
 - role of project manager in, 281–282
 - Physical distribution, 130
 - Pilot projects
 - adopting RUP in, 249
 - defined, 210
 - implementing major change, 218–220
 - implementing moderate change, 217
 - setting up, 213–214

- Pipeline organization, 8
- Pirsig, Robert, 363–364
- Planning. *See also* Project plans
 - adopting RUP, 198, 207
 - deploying RUP, 214
 - risk and, 134–135
 - role of developer, 352–353
 - role of project manager, 283
- Plug-Ins
 - adopting RUP using, 204
 - building RUP Process Configuration with, 180–181
 - configuration/process authoring tools and, 21–22
 - creating, 190–191
 - defined, 385
 - downloading, 182
 - Structural, 190, 192–196
 - Thin, 189, 191–192
- PMBOK (*Guide to the Project Management Body of Knowledge*), 277
- PMI (Project Management Institute), 277
- PR (Product Release Milestone), 11
 - overview of, 175–176
 - project plan including, 227
- PRA (Project Review Authority), 276
- Pressman, Roger, 273
- Prioritization, 295–296
- Private workspaces, 352
- Problem statement, Vision Document, 294
- Process and tool enhancement projects. *See* PTEPs (process and tool enhancement projects)
- Process authoring tools
 - defined, 20
 - overview of, 22
- Process Components
 - defined, 385
 - overview of, 180
 - RUP Modeler and RUP Organizer, 190–191
- Process Configuration. *See* Configuration (RUP Process Configuration)
- Process configuration tools, 20–22, 80
- Process delivery tools
 - defined, 20
 - overview of, 22–24
- Process Engineer, 328
- Process Engineering Toolkit, 191
- Process framework, 19–21
- Process improvement
 - continuous, 221
 - coupling with business results, 250–251
- Process Library, 191
- Process map
 - agile processes, 52–53
 - CMM/CMMI, 55–56
 - military standards of software development, 57
 - placement on, 62–63, 65
 - process comparison with, 51
 - RUP framework and configurations, 58, 61
 - understanding process improvements with, 50
- Process maturity, assessment framework, 54–55
- Process views
 - defined, 316, 385
 - MyRUP product providing, 22–23
 - overview of, 22
 - producing, 182–183
- Processes
 - breakdowns in, 223–224
 - concept of, 52
 - developing content of, 195–196
 - implementing in Inception, 108–110
 - outlining in Elaboration, 130
 - refining in Elaboration, 115, 136–138
 - role of project manager, 274–275
- Processes, comparing, 49–66
 - determining RUP configuration, 61–63
 - High Ceremony approach, 53–57
 - iterative needs and, 59–60
 - level of ceremony and, 58–61
 - Low Ceremony approach, 50–53
 - overview of, 49–50
- Product
 - breakdowns, 223
 - concept of quality, 360–361
 - defining RUP, 3–4
 - objectives of project manager, 274
- Product acceptance tests
 - defined, 168
 - overview of, 173–175
 - role of analyst in, 309
- Product Release Milestone. *See* PR (Product Release Milestone)



- Product Release Milestone (PR), 11
- Product releases. *See also* Beta release
 - comparing builds with, 351
 - defined, 385
 - Project Deimos, 80–81
- Programming
 - guidelines for implementing, 347
 - pair, 356–357
 - role of architect in, 327
- Project Deimos, 67–84
 - comparing project types, 63
 - describing, 61
 - getting commitment to, 75–77
 - making changes, 78–80
 - overview of, 67–68
 - proposal for, 69–74
 - seminal idea for, 68–69
 - shipping beta version, 81
 - sketching/completing product interface, 77–78
 - testing, 80–81
- Project Ganymede
 - comparing project types, 63
 - Construction deployment, 158
 - Construction progress, 150–151
 - describing, 61–62
- Project Ganymede, Elaboration
 - detailing requirements, 119
 - iterations and, 116
 - mitigating essential risks, 135
 - refining Development case, 136
 - testing executable architecture, 133
- Project Ganymede, Inception
 - costs, schedule and risks in, 108
 - critical use cases in, 104
 - determining architecture, 106
 - implementing process/tools, 109–110
 - iterations pattern in, 95
 - key actors and use cases, 101–102
- Project Genesis
 - motivating RUP implementation in, 200
 - planning for RUP, 201–203
- Project Jupiter
 - comparing project types, 63
 - deploying in Construction, 159
 - describing, 62–63
 - progress in Construction, 151
- Project Jupiter, Elaboration
 - detailing requirements, 120
 - iterations and, 116
 - mitigating essential risks, 135
 - refining Development case, 136–137
 - testing executable architecture, 134
- Project Jupiter, Inception
 - costs, schedule and risks in, 108
 - critical use cases in, 104
 - determining architecture, 107
 - implementing process/tools, 110
 - iterations pattern in, 96
 - key actors/use cases, 102
- Project management, 276–280
 - iterative development, 279
 - overview of, 276–277
 - risks, 279–280
 - scope of, 277
 - Software Development Plan, 278–279
- Project Management Institute (PMI), 277
- Project manager, 273–286
 - activities of, 280–282
 - architect working with, 313, 323–324
 - avoiding too many late changes, 261
 - estimation techniques, 239–241
 - finding way as, 282–283
 - iteration plans and, 228
 - mission as, 273–276
 - project management and, 276–280
 - resources for, 285–286
 - summary, 283–284
- Project Mars
 - comparing project types, 63
 - deploying in Construction, 158
 - describing, 62
 - progress of in Construction, 151
- Project Mars, Elaboration
 - detailing requirements, 119
 - iterations and, 116
 - mitigating essential risks, 135
 - refining Development case, 136
 - testing executable architecture, 133–134
- Project Mars, Inception
 - costs, schedule and risks in, 108
 - critical use cases in, 104
 - determining architecture, 107



Project Mars, Inception (*continued*)
 implementing process and tools, 110
 iterations pattern in, 96
 key actors/use cases, 102

Project Mercury
 motivating RUP implementation in,
 199–200
 planning for RUP, 201

Project plans, 223–242. *See also* Planning
 coarse-grain/fine-grain, 226–229
 developing iterations, 236–239
 estimation, 239–241
 iteration length and, 234–235, 259–260
 key concepts, 224–226
 motivation, 223–224
 number of iterations and, 232–234
 optimizing, 241–242
 overview of, 226–232
 paying lip service to RUP and, 252
 role of project manager, 278–279
 staffing profile, 235–236
 Wideband Modified Delphi and, 240–241

Project Review Authority (PRA), 276

Project reviews, 276
 Initial Operational Capability (IOC)
 Milestone, 159
 Lifecycle Objective Milestone, 110
 PRA (Project Review Authority), 276
 Product Release Milestone, 175
 Project Mars, 108, 135

Project Web site, 187–189

Proof-of-concept prototype, 324–325

Prototypes
 architectural, 318
 constructing architectural proof-of-concept,
 324–325
 defined, 385
 use-case, 305–307

PTEPs (process and tool enhancement projects)
 continuous improvement of, 221
 defined, 210
 implementing major change, 217–221
 implementing moderate change, 216–217
 phases of, 210–212
 pilot projects and, 213–214

Q

Quality
 concept of, 360–361
 conforming to standards, 364–365
 cost of, 362–363
 guidelines for, 46–47
 incrementally improving, 249, 367
 integration/system testing increasing, 155
 paradigms of, 361–362
 principles of approach to, 6
 quantification vs., 363–364
 Transition phase focus, 168
 “Quality by Design”, 47

Quantification, quality vs., 363–364

R

RAS (Reuse Asset Specification), 221, 356

Rational ClearCase, 148

Rational ClearQuest, 209

Rational Developer Network (RDN), 20, 182

Rational Process Workbench (RPW), 21, 190–191,
 385

Rational RequisitePro, 209

Rational Rose, 30

Rational Suite TestStudio, 209

Rational Unified Process. *See* RUP (Rational Uni-
 fied Process)

Rational XDE
 RUP Modeler as add-in to, 190
 Structural RUP Plug-Ins and, 194
 as tool mentor, 18
 unpackaging RAS assets, 356

RDN (Rational Developer Network), 20

Reading, resources
 analysts, 310
 architects, 329–331
 developers, 357
 project managers, 285
 testers, 380

Refactoring, 355

Regression testing
 focus of Transition, 168
 minimizing costs through, 155
 overview of, 47

Releases. *See* Product releases

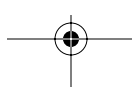
- Requirements
 - adopting RUP and, 246–247, 249
 - common mistakes, 252, 264–265
 - detailing in Elaboration, 115, 118–120
 - principles of approach to, 4, 7
 - progress in Construction, 143–144, 151–152
 - role of analyst, 288, 297–299, 307–308
 - role of architect, 323–324
 - role of developer, 336–337
- Resources
 - analysts, 310
 - architects, 329–331
 - developers, 358
 - project managers, 285–286
 - testers, 380–381
- Return on investment (ROI)
 - producing artifacts and, 34
 - tool automation and, 208
- Reuse
 - developer maximizing, 355–356
 - iterative approach to, 8
- Reuse Asset Specification (RAS), 221, 356
- Reviews
 - architectural, 327–328
 - code, 347–348
 - PRA (Project Review Authority), 276
- Rhythm, 321
- Risk
 - defined, 386
 - design completion in Construction and, 153
 - development phases and, 88–89
 - executable software and, 33
 - guidelines, 28–31
 - incremental improvements and, 249
 - iteration length and, 231
 - iterative approach to, 8
 - multiple iterations and, 95
 - pilot projects and, 213
 - principles of approach to, 4
 - project manager and, 279–280
 - understanding in Inception, 107–108
- Risk List
 - failing to handle difficulties on, 257–258
 - paying lip service to RUP, 252
 - Project Deimos, 72, 76, 78–79
- Risk mitigation
 - addressing, 115
 - adopting RUP and, 246–247, 249
 - critical use cases and, 125
 - iterations and, 116–118
 - objectives, 114–115
 - planning project and estimating costs, 134–135
- Roadmaps, 16
- ROI (return on investment)
 - producing artifacts and, 34
 - tool automation and, 208
- Roles
 - analyst, 309–310
 - architect, 319–320, 328
 - defined, 13, 386
 - developer, 357
 - Development case specifying, 186
 - major milestones and, 89
 - overview of, 13–14
 - project manager, 274–275
 - testing, 370–371
- Rollout
 - planning, 200–203
 - preparing for market, 173
- Royce, Walker, 242
- RPW (Rational Process Workbench), 20, 190–191, 385
- Runtime analysis
 - architect describing, 326–327
 - describing in Elaboration, 130
 - testing by developer, 348–349
- Runtime processes, 326–327
- RUP approach
 - iterative development and, 6–9
 - underlying principles of, 4–6
- RUP Base
 - defined, 386
 - RUP Process Configurations and, 180
- RUP Builder
 - adopting RUP in project, 204
 - defined, 386
 - importing plug-ins into, 195–196
 - overview of, 180–182
- RUP Exchange
 - defined, 386
 - downloading RUP Plug-Ins, 182

- RUP Glossary, 283
 - RUP Library
 - defined, 386
 - RUP Process Configurations and, 180, 182
 - RUP Modeler
 - as add-in to Rational XDE, 190
 - creating Structural Plug-Ins with, 192–196
 - defined, 386
 - overview of, 190
 - RUP Organizer
 - creating Structural Plug-Ins with, 192–196
 - creating Thin Plug-Ins with, 191–192
 - defined, 386
 - overview of, 191
 - RUP Process Configuration. *See* Configuration (RUP Process Configuration)
 - RUP Process Library, 191
 - RUP product, 19–25
 - adding extensions to, 19
 - companies using, 24–25
 - configuration and process authoring tools, 20–22
 - overview of, 19–20
 - process delivery tools, 22–24
 - RUP (Rational Unified Process)
 - adopting. *See* Adoption
 - approach, 4–9
 - configuration. *See* Configuration (RUP Process Configuration)
 - as customizable process product. *See* RUP product
 - defining, 3–4
 - iterative development approach, 6–9
 - plug-ins. *See* Plug-Ins
 - process framework, 19–21
 - resources. *See* Resources
 - software process. *See* Software engineering process
- S**
- SAD (Software Architecture Document)
 - critical use cases listed in, 104
 - Elaboration using, 319
 - overview of, 317–318
 - role of developer, 337, 356
 - Scenarios
 - defined, 387
 - implementing/testing critical, 131–133
 - Scheduling, 107–108
 - Scope
 - costs of changes to, 37
 - generating in Inception, 98
 - iteration length and, 231
 - iteration planning and, 236
 - multiple iterations and, 95
 - project management and, 277
 - Scrum, 51–52
 - SDP (Software Development Plan)
 - overview of, 278–279
 - project manager developing, 281, 283
 - SEEA (Software Engineering Environment Authority), 276
 - SEI CMM (Capability Maturity Model), 54–56
 - SEI CMMI, 55–56
 - SEPA (Software Engineering Process Authority), 276
 - Sequence diagram, 345
 - Simplification, architectural, 322
 - Smoke tests, 376–377
 - Software Architecture Document. *See* SAD (Software Architecture Document)
 - Software development approach
 - defining RUP, 3
 - development cycles, 165–167
 - guidelines for, 33–35
 - iterative development and, 6–9
 - underlying principles of, 4–6
 - Software Development Plan. *See* SDP (Software Development Plan)
 - Software development projects
 - deploying RUP in, 214–215
 - implementing major change, 219, 221
 - implementing moderate change, 217
 - Software Engineering Environment Authority (SEEA), 276
 - Software engineering process. *See also* Static Process Structure
 - Dynamic Process Structure, 10–13
 - executable software and, 33–35
 - purpose of, 67

- RUP, 3
 - use cases and, 32–33
 - Software Engineering Process Authority (SEPA), 276
 - Software Process Engineering Metamodel (SPEM), 9
 - SOW (Statement of Work), 97
 - Specifications. *See* Standards
 - SPEM (Software Process Engineering Metamodel), 9
 - “Spirit of RUP.” *See also* Guidelines
 - incremental improvements and, 249
 - overview of, 5
 - paying lip service to, 251–253
 - role of analyst in keeping, 301–302
 - Staffing. *See also* Teams
 - building project plan and, 228–231
 - planning for, 235–236
 - possible degrees of iteration, 233
 - project plan including, 227
 - Stakeholders
 - agreeing on what to build, 96–102
 - business focus of, 250–251
 - defined, 292
 - detailing requirements and, 265
 - documenting requests from, 292–293
 - failure to set right expectations for, 254–256
 - multiple iterations and, 95
 - partnering and, 322
 - role of architect, 314
 - Standards
 - conforming to, 364–365
 - high-ceremony software and, 56–57
 - Statement of Work (SOW), 97
 - Static Process Structure, 13–18. *See also* Software engineering process
 - activities, 14–15
 - additional process elements, 16
 - artifacts, 15
 - defined, 10
 - disciplines, 17–18
 - overview of, 13
 - roles, 13–14
 - workflows, 16
 - Steps
 - in activities, 15
 - defined, 387
 - Stereotypes, 340
 - Structural RUP Plug-Ins
 - adopting RUP in project, 204
 - creating, 192–196
 - defined, 190
 - Stubs
 - component emulation with, 348
 - developing, 353
 - Subsystems
 - completing design, 152–153
 - consolidating, packaging identified classes, 127
 - defining, 122–123
 - ensuring architectural coverage, 128–129
 - Supplementary Specification, 337
 - Synchronization, 132
 - System, identifying key functionality, 102–104
 - System testing, 154–155
- T**
- Targets, 224
 - Teams
 - avoiding functionally oriented, 253–254
 - guidelines for, 43–45
 - high-ceremony projects and, 54
 - incrementally adopting, 249
 - iterative approach and, 8–9, 59
 - multiple iterations and, 95
 - organizing around architecture, 145–147
 - pilot projects and, 213–214
 - principles of approach and, 6
 - progress and, 150
 - role of project manager, 274–276
 - software architect, 313
 - waterfall development process and, 6
 - Templates
 - adding, 18
 - customizing RUP, 184
 - defined, 16
 - Process Engineering Toolkit, 191
 - Terminology
 - creating glossary for project, 100
 - glossary for book, 383–387
 - Test approach, 376
 - Test assets, 378–379
 - Test automation architecture, 373
 - Test cases, 82, 372



- Test cycles
 - defined, 378–379
 - overview of, 369
 - role of tester, 377
- Test evaluation summary, 371
- Test-first design, 52, 355
- Test-idea list
 - overview of, 371–372
 - RUP test philosophy and, 368
- Test interface specification, 373
- Test plans, 371
- Test scripts, 372
- Test stubs. *See* Stubs
- Test suite, 372
- Testability, 373
- Testers, 359–381
 - activities of, 373–379
 - defining testing, 365–367
 - key test artifacts, 371–373
 - mission of, 359–365
 - resources for, 380–381
 - RUP testing philosophy, 367–370
 - various roles of, 370–371
- Testing
 - adopting RUP and, 247, 252
 - assessment and, 269–270
 - beta, 162, 167–169, 174
 - critical scenarios, 131–133
 - defining, 365–367
 - guidelines for, 46–47
 - necessity of, 59
 - performance and, 122
 - Project Deimos, 80–81
 - role of developer, 347–349, 354–355
 - use cases in Elaboration, 118–119
- Testing, in Construction
 - emphasis, 141
 - ensuring progress through, 150
 - implementations, 153–154
 - iterations, 143–144
 - system, 154–155
- Testing, in Transition
 - metrics for, 170–171
 - overview of, 168
 - product acceptance and, 173–175
 - role of analyst, 309
- Thin RUP Plug-Ins
 - adopting RUP using, 204
 - creating, 191–192
 - defined, 189
- Threads, 130
- Time-boxing
 - activities on use cases, 101
 - Inception phase objectives and, 98, 101
 - overview of, 225–226
- Tool automation
 - achieving ROI through, 198, 208–209
 - efficiency of development and, 60–61
 - identifying tested code through, 348
 - implementing in Inception, 109–110
 - iterative approach and, 59
 - testing critical scenarios, 132–133
 - testing making insufficient use of, 366–367
- Tool mentors
 - adding to process, 18
 - defined, 16, 387
 - overview of, 23–24
 - planning process and tool environment, 202–203
- Tools. *See also* PTEPs (process and tool enhancement projects)
 - configuration and process authoring, 20–22
 - customizing environment for, 204
 - evaluating effectiveness of, 206–207
 - implementing in Inception, 109–110
 - planning for, 200–203
 - process delivery, 20, 22–24
 - testing making insufficient use of, 366–367
- Training
 - adopting RUP and, 200–203
 - customization and, 204
 - final deployment and, 157
 - iterative approach and, 8–9
 - outlining RUP project and, 205–206
 - users/maintainers, 171
- Training resources
 - analysts, 310
 - developers, 357
 - testers, 381
- Transition phase, 161–176
 - artifacts and, 90–91
 - beta testing in, 167–168, 169



- construction and, 237–238
- converting operational databases, 172
- costs of changes during, 36
- defined, 13, 387
- deployment site preparation, 172
- development cycle, 165–167
- future product improvement and, 175
- iterations and, 163–165, 232–234
- launch-packaging preparations, 172–173
- market rollout preparations, 173
- metrics for, 169–171
- misconceptions about, 87–88
- multiple bids, 255–256
- objectives/milestones of, 11, 162–163
- overview of, 161–162
- patch/beta releases, 169
- process and tool enhancement projects at, 212
- product acceptance testing, 173–175
- Product Release Milestone of, 175–176
- Project Deimos and, 72
- risk mitigation in, 88–89
- role of analyst in, 309
- role of architect in, 320
- role of tester in, 375
- testing in, 168, 375
- training users, 171
- workflows, 89–90

U

- UI (User-interface) prototypes
 - complementing use-case descriptions with, 301
 - developer reviewing, 336
 - developing, 304–307
 - risk reduction using, 30
- UML models, 307
- Unit testing, 153–154
- Use case driven development, 31–32
- Use case models
 - example specification, 302–303
 - fine-tuning, 304
 - guidelines for detailing, 300–302
 - role of analyst in developing, 296, 298–299
- Use-case prototypes, 305–307
- Use-case realizations. *See also* Design, use cases/
 - components
 - adopting RUP incrementally, 247
 - overview of, 125–127

- Use case views, 317
- Use cases. *See also* Design, use cases/components
 - activities in Construction, 151–152
 - brainstorming sessions identifying, 100
 - common mistakes, 252
 - creating too many, 262–263
 - defined, 98, 387
 - detailing, 101–102, 118–119, 299–302
 - development choices, 91
 - developer implementing, 347
 - developing use-case model, 296–299
 - driving architecture with, 123–129
 - generating project scope, 98
 - identifying key system functionality, 102–104
 - iteration planning driven by, 142–144
 - prioritizing, 324
 - requirements of, 264, 336–337
- Use patterns, 355–356
- The User-Experience Modeling Plug-In for the RUP*, 307
- User-interface prototypes. *See* UI (User-interface)
 - prototypes
- Users
 - avoiding many late changes for, 261
 - beta deployment and, 157
 - delivering value to, 31–33
 - early deployments/feedback loops and, 156
 - generating project scope, 98
 - preparing for unexpected interactions of, 133–134
 - training for self-reliability, 171
 - Transition beta testing and, 167–168
 - Transition focus on, 162
 - Transition iterations and, 165
- V
- Value
 - guidelines for, 31–33
 - incrementally improving, 249
 - principles of, 4
- View of Participating classes (VOPC), creating, 340–341
- Views
 - Process Views, 22
 - types of architectural, 316–317
- Vision
 - adopting RUP and, 247



Vision (*continued*)

- agreeing on high-level, 96–97
- altering, 91
- architectural, 320–321, 324
- defined, 387
- developing, 293–296
- producing document, 97
- Project Deimos, 69–71, 75, 79–80
- role of project manager, 281
- Vitruvius, 311–312
- VOPC (View of Participating classes), creating, 340–341
- VRAPS model, 320–322

W

- Waterfall development
 - CMM encouraging, 55
 - defining, 6
 - failure to adopt RUP and, 252
 - focus on inspections in, 269–270
 - iterative approach vs., 6–9, 59–60, 87–88
 - Low Ceremony/High Ceremony approach vs., 50–51
 - risk reduction profile for, 29

testing approach of, 367–368

- Transition vs., 162
- wrong expectations leading to, 255

Web resources

- architects, 331
- project managers, 285–286

Web services, 42

Web site, project, 187–189

Wideband Modified Delphi, 240–241

Workflow

- defined, 13, 387
- not fixing, 89–90
- overview of, 16

Workflow Details, 16

Workload model, 373

Workshops, 98–99

Workspaces

- configuration management, 352
- pair programming and, 356–357

X

XP (Extreme Programming)

- as agile process, 51–52
- comparing RUP architecture with, 268

