Index

A
academics, and quality topics, xxviii–xxix, xxx
acceptance criteria, in testing, 335–336
acceptance test, 316
activities and tasks in
  business evaluation, 166–167
  change management, 202
  deployment, 212
  enterprise architecture, 205
  implementation, 193, 194
  interface modeling, 182, 183
  persistence design, 190
  process configuration, 174
  project management, 170, 171
  prototyping, 198, 199
  quality assurance, 225
  quality control, 227, 228
  quality management, 220, 222
  requirements modeling, 177, 178
  reuse, 218
  system architecture, 209
  system design, 186
  training, 215
activity diagrams, as deliverable, 177–178
activity element in a process, defined, 159
Adams, Scott, 121
administrative procedures, required in testing, 335
adoption, of a software process. *See* deployment
Adult ego state, 109, 115. *See also* ego states
aesthetics, quality assurance checks, 44–45
agendas, antidote for Meetingitis, 121
Ahlgrin, M., 268
Alexander, Christopher, 4–5
analysts. *See* business analyst; quality analyst
analyzing results, 338
analyzing risks in testing, 319–321
anti-virus software, 318
architects. *See* enterprise architect; system architect
architecture
  in background space, 32
  testing, 313–314
artificial crisis, creating, 235
audits
  compared to inspections and reviews, 127
  described, 127–128
  documentation of results, 128
  internal and external, 127
  and malleability, 123
  written objectives required, 128

automated testing, 307, 309–311
awkward silence, in workshops, 132

B
background space. *See also* model of the background space; team organization, roles in background space
  defined, 32–33
  organizing the roles, 93, 94
baking example
  creating a process-component, 161, 162
  measuring, 277, 278–279
  process, 20
barber, 10
Beizer, Boris, 303
Berne, E., 116
best-fit approach, creating a homogeneous team, 107–109
black box testing, 306–308
Boehm, Barry, 239, 284
Booch, G., 56
boundary value testing, 307, 313
budgets
  in large projects, 68
  as quality pressure, 12
  for software maintenance, 12
bugs. *See* software, incidents
bureaucratic processes, and malleability, 52
business analyst
  attributes of, 85
  documentation abilities, 85
  and quality topics, xxviii–xxix
  as requirements modeler, 86
  responsibilities, 85
  role in
    implementation, 193
    interface modeling, 182
    requirements modeling, 177
    reuse, 218
  role on team, 85
business case, as deliverable, 166, 167
business evaluation process-component. *See* process-component for, business evaluation
business sponsor. *See* sponsor

C
Capability Maturity Model. *See also* process maturity
  applying, 141–142
  impact on quality and productivity, 147
  levels, 139–140
  process areas, 140
  process standards, 134–135
  role in processes, 138–140
CASE tools, keeping implementations separate, 251–252
Catalysis, as off-the-shelf software, 23
CDs, as communication mechanism, 105
change-control mechanism, for test cases, 334
change management process-component. *See* process-component for, change management
checklists
  creating, 128
  described, 128–129
  dynamicity, 128–129
  paper or electronic, 128
  should be graded, 128
  used in walkthroughs, 123–124
Child ego state. *See* ego states
class attributes and operations, 156
class testing, vs. use case testing, 327
classes, measuring size and complexity, 287
closed teams, 106–107, 116
closed testing. *See* black box testing
CMM. *See* Capability Maturity Model
CMM Integration production suite (CMMi), 138–139
COCOMO model, 284
code (executable), as deliverable, 194
code quality, 16
coder. *See* programmer

coffee break, value in workshops, 132
commercial off the shelf. *See* off-the-shelf software
common roles, 95
communication
  facilitated by UML, 34
  facilitating by models, 28
  facilitating through standards, 137
  in outsourcing projects, 61
  in quality environment, 104–105
  responsible for project failures, 17
  suggested mechanisms, 104–105
  and team structure, 110
competence, individual, and best-fit, 109
component test, 315
components, measuring size and complexity, 287
components, reusable. *See* reusability
conferences
  IT Project Management conference, 17
  OOPSLA 95, 35
  OOPSLA 96, 4, 34, 81
  TOOLS 2000, 28
  UML 2001, xxix
confusion, value of, 132
Constantine, Larry, 54, 55, 79, 81
construction, quality of, 41
context diagram, display of, 105
conversion plan, as deliverable, 190
conversion projects, 58, 62, 309
cooking process, sample. *See* baking example
copyrights. *See* legal issues
corporate head office, and standards, 137
costs. *See* budgets
COTS. *See* off-the-shelf software
Cowboy Programming game, 119–120
creativity, and Cowboy Programming, 120
creep factor. *See* scope creep
CRMS. *See* Customer Relationship Management System
Crystal, as off-the-shelf software, 23
cumbersome processes, and malleability, 52
Customer Relationship Management System, 60
cyclic approach to testing, 323

D

data modeler, 91
data, probabilities of error in, 340
data, quality, 15
data warehousing projects
  UML, 58, 62
  white box testing, 309
database design, as deliverable, 190
database designer, role in persistence design, 190
database, for test results, 338
database manager
  interaction with data modeler, 95
  role in persistence design, 190
  role on team, 95
  testing role, 321
databases. *See also* data modeler
  and measurements, 269
  model in data warehouse / conversion projects, 62
  standards, 135
  and test plan, 319
  for test results, 319
Deadline game, 121–122
deadlines, missed, 12
deliverables
  business evaluation, 167
  change management, 202, 203
  defined, 160
  deployment, 212
  enterprise architecture, 206
  implementation, 194
  interface modeling, 182, 183
  persistence design, 190
  of planning. *See* planning deliverables
  process configuration, 175
  project management, 170
  prototyping, 198
  quality assurance, 225, 226

deliverables (*cont.*)
quality control, 227, 228, 229
quality management, 220, 221
requirements modeling, 177–178
reuse, 219
system architecture, 209
system design, 187
training, 216
DeMarco, T., 267
deployment, 249–250, 251, 316–317
deployment plan, as deliverable, 212
deployment process-component. *See* process-component for, deployment
depth of quality checking. *See* sufficiency
designers. *See* database designer; interface designer; system designer
developer, and quality topics, xxviii–xxix
development environment, 322
development projects, 58, 81
Dilbert, 121
dimensions of a process, 152, 250
director. *See* senior management
disturbances (physical), impact on team. *See* e-factors
documentation
in test acceptance, 336
and UML, 42
dog, learning to croak, 47
domain expert, 88, 89, 177
downloads. *See* legal issues
driving conditions, 257

E

e-factors, 101, 102
educational projects, 57, 62–63
ego states
characteristics, 115
and games, 117
impact on games, 118
in transactional analysis, 114–115
elegance, of a process, 51
email, as communication mechanism, 104
enactment process-component. *See* process-component for, enactment
encapsulation. *See* object-oriented
end user
distinct from user, 87
helps with global issues, 88
importance to defining requirements, 88
and modeling, 28
and quality, 7
role on team, 87
testing role, 321
engineers. *See* process engineer
enhancements, as software incidents, 337
enterprise architect, responsibility for reuse, 94
enterprise architecture process-component. *See* process-component for, enterprise architecture
Enterprise Resource Planning, 60
enterprise wide architecture, as deliverable, 206
environment
development, 322
physical. *See* e-factors
testing, 322
working, and tight estimates, 275
equation, measuring object-oriented systems, 292
equivalence partitioning, 307, 312–313
ergonomics. *See* usability
ERP. *See* Enterprise Resource Planning
errors
costs of correction, 25
introduced by fixes, 317
test acceptance, 336
estimates
about, 268
initial, techniques, 283
in Lucky Insurance project, 290–291, 294–295
measurement of, 269, 270–271

measurements and estimates, compared, 268–269
project, 271–274
refining at end of iterations, 279–282
relating estimates to quality, 268
role in quality assurance, 268
tight, effects of, 275
estimates and measures. *See* measurements
estimating process, defined, 268
existing applications. *See* legacy
experimental projects. *See* pilot projects
expert experience and knowledge, reusing, 53
experts. *See* domain expert
external audits, when needed, 127
extraneous factors. *See* road factors
eXtreme Programming
in development projects, 58
and games, 120, 122
as off-the-shelf software, 239
and process-components, 232
Eykolt, Ed, 56

F
facilitator, in workshops, 131
facilitators, 110
FDD, as off-the-shelf software, 23
Flour Mix game, 120
flow of information. *See* communication
flowcharts, as a visual model, 40
formula, measuring object-oriented systems, 292
fountain-based SDLC, 242
Fowler, M., 34
function point estimation, 284–285
functional prototype, as deliverable, 183
functional specifications, as deliverable, 177–178
functionality, adding, 13, 120

G
games, 117–122
Gamma, Erich, 53
Gang of Four, 53
garbage collection, 316
Gates, Bill, 13
global issues, 88, 137
glossary of business terms, as deliverable, 178
Go or No Go decision, 164, 316
Goldberg , A., 84
Graham, I., 112
granularity, of object-oriented systems, 286
graphical user interface. *See* GUI; interfaces
GUI, 56, 340. *See also* interfaces

H
hardware, needed in testing, 320
Harry Potter, 49
hierarchical team structure, 109–111
high-visibility projects, 65–66
horizontal testing, 307, 311
"how" of a process. *See* methodological aspects
human factors. *See* sociological aspects; usability

I
ICONIX, as off-the-shelf software, 23, 239
IIP development process
development plan creation, 145
integration plan creation, 145
iterations and increments, 243–249
project plan creation, 264
project plan, section, 256
ill-fitting individuals, 109
I'm [not] OK, You're [not] OK. *See* life positions
implementation process-component. *See* process-component for, implementation
implementer. *See* programmer
incidents, software. *See* software, incidents
increments. *See* iterations
India, and IT industry, 146

information flow. *See* communication
informative events, as software incidents, 337
initial cycles reports, 341
inspections
  described, 124–125
  documentation of results, 124
  by peers, 124
  of quality models, 47
  usability, 124–125
  in white box testing, 308
integration projects, 57, 58–59
intellectual property. *See* legal issues
interface design, as deliverable, 183
interface designer, 91
interface modeling process-component. *See* process-component for, interface modeling
interfaces. *See also* GUI
  legacy, as deliverable, 187
  modeling and design. *See* process-component for, interface modeling
  navigability of, 55–56
  shared, social issues, 103
  specifications, as deliverable, 182
  system, not just GUI, 9
interviews, 129–130
intranet, 104, 138
IT Project Management conference, 17
iterations
  creating, 253–255
  defined, 161
  deploying process-components, 245
  effort per iteration, 244
  final, 248, 262–263
  initial, 245–246, 260–261
  maintenance or ongoing, 249
  major
    described, 246–248
    quality activities at end, 261–262
    and sociological aspects, 280–281
  measurements based on initial, 294
  need for, 243
  parallel development, 248–249
  refining estimates after each, 279–282
Iterative, Incremental, Parallel. *See* IIP; iterations
iterative project management tools, 256–257
iterative project task plan, 255–256

J
Jacobson, I., 35, 151, 285

K
Kimball, R., 137

L
large projects, 67–69
lectures, on quality topics, xxx
legacy
  applications, in integration projects, 59
  interfaces, as deliverable, 187
  systems, helped by modeling, 72
legal issues, 103, 127
levels of process maturity. *See* process maturity
levels, of quality process, 50
life positions
  affect on teams, 116
  defined, 116
  impact on games, 118
  and team structure, 116, 117
lifecycle, parallel development, 248–249
Lim, W. C., 111
lines of code, as software measurement, 284, 285
Lister, T., 267
LOC. *See* lines of code
Lockwood, L., 54, 55
logic validation, in test acceptance, 336
Lucky Insurance project
  applying measurements and estimates to, 287–297
  configuration of processes, 238
  iterations, 243, 253–255

M
major errors, risks of discovering late, 320
make vs. buy, as background space issue, 33
malleability
  aspect of baking example, 20
  aspect of process, 164
  contribution to quality, 52, 144
  defined, 51–52, 144
  of process-components, 143–144
  and process discipline, 263
  and process engineer, 99
  revealed by audit, 123
management
  in background space, 32
  need for reward system for reuse, 112
  of overall quality function, 27
  problem and solution space, 32
  reward system for reuse, 111
  sociological aspects, 80
management of quality vs. quality of management, 17–18
managers. *See* database manager; project manager; quality manager; test manager
manual testing, 306, 309–310
Marick, Brian, 324
maturity. *See* process maturity
measurements
  of classes, 287
  of components, 287
  deciding what to measure, 283
  defined, 268
  importance of, 297
  in Lucky Insurance project, 287–297
  and multipliers, 278
  of processes, 275–282
  of project size and type, 271–272
  in projects. *See* project measurements
  of quality models, 282–283
  of resource distribution, 272, 273, 274
  role in quality assurance, 268
  of software, 284–286
  stored in database, 269
  of UML elements, 286–287
  and weighting factors, 278
medium projects, characteristics, 66–67
Meetingitis game, 121, 131
memory leaks, 316
MeNtOR, as off-the-shelf software, 23, 239
mentoring and training, 252
metamodels
  class attributes and operations, 156
  creation of, 37
  levels of, 37–38
  to prevent modeling errors, 38
  of a process. *See* process metamodel
  and quality, 37–38
  for UML, 36
methodological aspects
  of baking example, 20
  measurement of, 269
  of a process, 152, 154
methodology war, 35
metrics. *See* estimates; measurements
Meyers, Glenford J., 339
MOBS. *See* model of the background space
model of the background space, 93–94, 226. *See also* modeling spaces
model of the problem space, 84, 225. *See also* modeling spaces
model of the solution space, 89, 225. *See also* modeling spaces
model quality, 16
modeler, role in quality control, 227
modelers. *See* data modeler
modeling, 17, 27–29
modeling and quality, 27–29
modeling spaces. *See also* background space; problem space; solution space
  importance of separation, 71
  interdependencies, 33
  and UML diagrams, 42–44
  understanding, 29–31

models
  quality of, 44–45
  testing validity, 308
  and UML diagrams, 44
  views of, 30
MOPS. *See* model of the problem space
MOSS. *See* model of the solution space
multipliers, in measuring, 278

N
navigability of interfaces, 55–56
necessity, 50–51, 142–143
network, needed in testing, 320
newsletters, as communication mechanism, 105
Next Release Maybe, 120
nontechnical management, 80–82
notations for describing process ingredients, 158
NRM. *See* Next Release Maybe

O
Object Management Group, 34, 37–38
Object Oriented Programming, Systems, Languages, and Applications. *See* OOPSLA
object-oriented software, is not an object, 8
object-oriented systems
  affect on programmers, 92
  encapsulation, improves quality, 53
  measuring granularity, 286
  objects, success tied to business contexts, 84
  parallel development, 112
  resurgence of, 113
  sociological aspects, 113
  software measurements, 285–286
objectives
  high level, display of, 105
  lack of, 17, 121
off-the-shelf software, 23, 237–239
office tools, impact on team. *See* e-factors
OK position, 109. *See also* life positions
Olympic trainer, 10
OOPSLA 95, 35
OOPSLA 96, 4, 34, 81
OPEN Consortium, as off-the-shelf software, 23, 239
open teams, 106, 107
open testing. *See* white box testing
operational plan, as deliverable, 212
operational requirements, in test acceptance, 336
operational testing, 317
orthogonal process relationship, 21–22
Osho, 71
outside factors. *See* road factors
outsourcing projects, 58, 61–62

P
package implementation projects, 57, 60–61
parallel team organization, 102, 112–113
Parent ego state. *See* ego states
parking lot
  antidote for Meetingitis, 121
  in workshops, 132, 133
partitioning testing approach, 307
pattern libraries, as deliverable, 187
people. *See* staff
PeopleSoft, 60
performance testing, 318
Perry, William, 5, 26
persistence design process-component. *See* process-component for, persistence design
personalities, 126, 141. *See also* ego states; sociological aspects
phone. *See* telephone
physical arrangements, impact on team. *See* e-factors
physical security, 318
pilot projects, 66, 250–251
planning deliverables
  project organizational plan, 144
  quality plan, 145
  test plan, 145–146
play acting, in workshops, 132
pressure, working under, 122

problem space. *See also* model of the problem space; team organization, roles in problem space
  activities, 31
  defined, 31
  focus on functionality, 89
  legacy applications, 31
  organizing the roles, 84
  prototype, impact on usability, 89
problems, as software incidents, 337
process
  configuration, 237–243
  defined, 19–20
  deployment, 236
  dimensions, 152–155, 250
  elegance, 51
  elements
    access to, 253
    defined, 157
    describing ingredients, 158
    shareability of, 253
  enactment
    importance of road factors, 257–260
    practicality of, 259
    in practice, 237
  inherent characteristics. *See* malleability
  iterations, 243
  malleability checks, 143–144
  maturity. *See* process maturity
  metamodel. *See* process metamodel
  necessity checks, 142–143
  in practice, 236
  process quality checks, 142
  and quality, 17, 82–83
  relationships, orthogonality, 21–22
  and reusability, 111
  rigorous, 163
  in software context, 22–23
  state, ascertaining current, 250
  sufficiency checks, 143
  timings, 164, 231
  why measure, 276
process-component for
  business evaluation, 165–168
  change management, 201–204
  deployment, 211–214
  enactment, 277–279
  enterprise architecture, 204–207
  implementation, 192–196
  interface modeling, 181–184
  persistence design, 188–192
  process configuration, 172–176
  project management, 169–172
  prototyping, 196–200
  quality assurance, 223–227
  quality control, 227–231
  quality management, 220–223
  requirements modeling, 176–181
  reuse, 217–220
  system architecture, 207–211
  system design, 185–188
  training, 214–217
process-components
  in baking example, 161
  for business evaluation, in Lucky Insurance project, 290
  defined, 21, 160
  for deployment, measuring, 276–277
  for enactment, measuring, 289
  evolution of, 236
  measuring, 275–282
  as process building block, 157
  of quality process, 220
  for requirements modeling, in Lucky Insurance project, 290
  and software engineering process, 237
process configuration process-component. *See* process-component for, process configuration
process consultant. *See* process engineer
process discipline. *See* process maturity
process engineer
  and CASE tools, 99
  as consultant, 252
  creation of software engineering process, 263

process engineer (*cont.*)
malleability of quality process and, 99
as process mentor, 252
and quality topics, xxviii–xxix
ready-made processes, 239
role in enacting, 236
role in process configuration, 173–174
role on team, 99
software engineering process, 237
process maturity. *See also* Capability Maturity Model
applying CMM in UML-based projects, 141–142
Capability Maturity Model, 138–141
defined level (level 3), 139–140
five levels, 139–140
identifying current state, 139
initial level (level 1), 139, 140
measured level (level 4), 140
measuring, 163
optimized level (level 5), 140
personal software process maturity, 141
and process dimensions, 141
repeatable level (level 2), 139, 140
process metamodel
activities, 159
deliverables, 160
described, 156–157
importance of understanding, 232
iterations, 161
process-components, 160
process-components, in baking example, 161–162
roles on team, 157–158
tasks, 159–160
process-thinker, role in enacting, 236
processes
didactic, contributes to Cowboy Programming game, 120
and malleability, 51–52
measurement of, 275–282
procurement of hardware and software. *See* project organizational plan
productivity
factor, in Lucky Insurance project, 291–294
increased by quality process, 25
increased through reuse, 53
nonlinearity with hours spent, 102
program director, role on team, 96
programmer
as cowboy, 119–120
role
described, 92
in implementation, 192
in quality control, 227
in reuse, 218
in system design, 186
on team, 92
as superhero, 119
programming language
features, in Use It or Lose It game, 119
standards, 119, 135
project brief, as deliverable, 166, 167
project management
responsible for project failures, 17
and telecommuting, 105
tools, iterative, 256–257
project management process-component. *See* process-component for, project management
project manager. *See also* program director
discouraging games, 119, 120
estimates, 283
as facilitator, 110
latest demo, 112
new sign-off paradigm, 96, 242
project organizational plan, 144–145
quality topics, xxviii–xxix
ready-made processes, 239
recruiting, 107
responsibilities of, 95, 112
role in
business evaluation, 166
change management, 202
deployment, 211

enacting, 236
enterprise architecture, 205
process configuration, 174
project management, 170
prototyping, 198
quality assurance, 224
requirements modeling, 177
reuse, 218
risk management, 95, 96
system architecture, 208
system design, 186
role, on team, 95
steering committee, 96
training in IIP approach, 252
project measurements, 267–268, 271–272
project objectives. *See* project organizational plan
project office, role in projects, 146–147
project organizational plan, 144–145, 170
project plan, separate from project organizational plan, 144
project sponsor. *See* sponsor
project task plan, 170, 255–256
project team
role in change management, 202
role in deployment, 212
role in enterprise architecture, 205
role in process configuration, 174
role in quality assurance, 224
role in training, 215
projects. *See also* UML-based projects
distribution of resources, 272–274
estimates and measures. *See* estimates and measures
failure of, 260–261
four team models, 106, 107
goals and objectives, as communication mechanism, 105
resources, measurements, 272–274
roles of developers and testers, 321
sociology, 106–113
soft factors, 102
prophetic statements, 296
prototype
architectural, as deliverable, 198, 209
evolve or throw away, 93
functional, as deliverable, 198
interface, as deliverable, 198
problem space, impact on usability, 89
technical, as deliverable, 187, 190, 198
prototyper
in background space, 95
in problem space, 89
role, 198
in solution space, 93, 95
prototyping process-component. *See* process-component for, prototyping
Putnam model, 284
pyramid, flattening, 109–111

Q

QA. *See* Quality Assurance
QAIindia, 141
QC. *See* Quality Control
quality
activities, 260–263
assuring, as a distinct activity, 10–11
and common sense, 7
of construction, and UML, 41
defined, 4, 5, xxiv
of documentation, and UML, 42
elusive nature of, 268
emotional nature of, 4
essential ingredients of, 6
and estimates, 268
extreme, impact on productivity, 8
GUI design, 56
importance of standards. *See* standards
of management, vs. management of quality, 17–18
measures of, 282–283
and metamodels, 37–38
of models, syntax, semantics, and aesthetics, 44–45
as a moving target, 7
and objective effort, 7–8

quality (*cont.*)
pressures. *See* quality pressures
priorities, 11
process. *See* quality process
relationship with estimates, 268
of specification, and UML, 40–41
and staff experience, 103
tracking, 257–263
and UML, 34, 38–39
of UML, vs. quality by UML, 35–36
underlying motto of, 7
and usability, 54–56
of visualization, and UML, 39–40
quality analyst
check on process checks, 99
role in
inspections, 124
modeling spaces, 99
quality assurance, 224
quality management, 220
testing, 321
role, on team, 98
quality assurance
as different from quality control, 26
influence on testing, 305
package implementation projects, 60–61
responsibilities, 304–305
of software process
described, 49–50
quality of process, 50–52
techniques
goals, 44–45
quality models, 45–49
and testing, 26–27
quality assurance process-component. *See* process-component for, quality assurance
quality comments on
business evaluation, 167–168
change management, 202–204
deployment, 213–214
enterprise architecture, 206–207
implementation, 194–196
interface modeling, 183–184
persistence design, 191–192
process configuration, 175–176
project management, 170–172
prototyping, 198–200
quality assurance, 226–227
quality control, 229–231
quality management, 222–223
requirements modeling, 179–181
reuse, 219–220
system architecture, 210–211
system design, 187–188
training, 216–217
quality context triangle, 15
quality control, 26, 304–305
quality control process-component. *See* process-component for, quality control
quality environment
communication in, 104–105
crucial aspect of quality process, 151
defined, 101
described, 18–19, 80
e-factor and quality, 101–102
and quality management, 80
and soft issues, 102–104
and telecommuting, 105–106
quality game, key points, 69–70
quality levels
code, 16
data, 15
described, 14–15
management, 17–18
model, 16
process, 17
quality environment, 18–19
quality management. *See also* software quality
areas of responsibilities, 304–306
described, 80
distinct from project management, 83
focus of, 97
influence on testing, 305
need for independence, 97
nontechnical management, 80–82
process, 82–83
project management, 72, 83, 97

quality environment, 80
responsibilities of, 304
and testing, 304
quality management process-component. *See* process-component for, quality management
quality manager
focus on soft factors, 98
project manager, 98
quality environment, 18
quality topics, xxviii–xxix
ready-made processes, 239
role in
change management, 202
enterprise architecture, 205
interface modeling, 182
persistence design, 190
project management, 170
prototyping, 198
quality assurance, 224
quality control, 227
quality management, 220
reuse, 218
system architecture, 208
system design, 186
role, on team, 97
steering committee, 97
training in IIP approach, 252
quality models
aesthetics, 48–49
semantics, 47–48
syntax, 45–46
quality plan, 145, 220
quality pressures
budget, 12
functionality additions, 13
impact on quality efforts, 13–14
quality priorities, 11–12
time, 12
quality process
architecture, described, 151–152
as aspect of quality environment, 151
described, 24–25
increase in productivity, 25
independence, 24
levels of software process maturity, 50
necessary, sufficient, and malleable aspects, 82
normal development process, 25–26
as part of quality software process, 163
as part of the quality software process, 24
process-component. *See* process component, for individual components
software models, 24
the software process, 22–23
software process, 24
software product, 24
quality software process. *See also* software process
of baking example, 20
defined, 23
defining process, 19–20
as deliverable, 175, 221, 225
as distinct from software process, 162–163
enacting, 235–237
malleability, 164
maturity, 163–164
orthogonal process relationship, 21–22
quality assurance and testing, 26–27
and the quality process, 24–26
quality process, 163
rigor of, 163
and software development, 23–24
and the software process, 22–23
timings, 164
quality team, 97, 220
quality techniques
audits, 127–128
checklists, 128–129
described, 122–123
inspections, 124–125
interviews, 129–130
mapping among, 122
quality of process and models, 123
reviews, 125–127

quality techniques (*cont.*)
walkthroughs, 123–124
workshops, 131–133
quality topics
organizing workshops, xxx
and team roles, xxviii–xxix
questionnaires, for interviews, 129

R
random teams, 106, 107
Rational Unified Process
as off-the-shelf software, 23, 239
and quality assurance, 10
recording (audio). *See* tape recording
recruitment, focus on best fit, 109
regression testing, 310, 317
released product, as deliverable, 212
report design, and interface designer role, 91
report designer, role on team, 91
requirements modeling process-component. *See* process-component for, requirements modeling
response, system, 318
results validation, 336
reusability
aspects of test plan, 324
correlation with quality, 111
of expert knowledge and experience, 53
and the increase in productivity, 53
and object-orientation, 52–53
and people, 111–112
reward system, 111, 112
social issues, 103
staff experience, 103
and standards, 53–54
woven into process, 111
reusable entities, as deliverable, 219
reusable libraries, as deliverable, 194
reuse
plan, as deliverable, 187, 219
strategy, 209. *See also* project organizational plan
reuse process-component. *See* process-component for, reuse
reverse modeling, 67–68
reviews, 125–127
reward system, for reuse, 111, 112
rigorous process, 163
risk analysis, 319–321
risk management
importance of project manager, 95, 96
in testing, 319–321
risks (project). *See* project organizational plan
road factors
and accurate estimates, 283
defined, 257–260
in estimates, 275
importance in process enactment, 257–260
in initial estimates, 283
role element in a process, 157
roles in
business evaluation, 166
change management, 202
deployment, 211, 212
enterprise architecture, 205
implementation, 192, 193
interface modeling, 182
persistence design, 190
process configuration, 173–174
project management, 170
prototyping, 198
quality assurance, 224
quality control, 227
quality management, 220
requirements modeling, 177
reuse, 218
system architecture, 208
system design, 186
training, 215
roles, swapping for testing, 321
Rowling, J. K., 49
royalties. *See* legal issues
Rubin, K., 84
rules of thumb, for estimates and metrics, 296

S
SAP, 60
scalability, 63–64, 318
scale of projects and application of UML, 64–65
schedules, affect of missed deadlines, 12
scope creep, 13, 120
scope of testing, 319–321
SDLC, types of, 241–242
security testing, 318
semantics, quality assurance checks, 44–45
senior management
  games, 118
  initial estimates, 283
  initial iteration, 260
  quality topics, xxviii–xxix
  steering committee, 96
  support for object-oriented techniques, 103
  testing results, 341
  time pressure, 12
shareability, of process elements, 253
short-term roles, 109
Siebel, 60
sitting arrangements, impact on team. *See* e-factors
slices, vertical and horizontal, 325
slicing testing approach, 307
small projects, characteristics, 65–67
sociological aspects
  of baking example, 20
  during major iteration, 280–281
  measuring, 271–272
  of object-oriented systems, 113
  of a process, 153, 155
  and quality, 81
  and reviews, 126
  of white box testing, 309
soft factors
  legal issues, 103
  often neglected, 147
  and parallel team organization, 102
  and reusability, 103
  senior management support, 103
  specific to UML-based projects, 102
  staff experience, 103
soft issues. *See* soft factors
software
  development lifecycle. *See* SDLC
  failing due to sociological aspects, 9
  incidents, 337, 339, 340
  maintenance costs, 12
  measurement. *See* measurements, of software
  modeling, issues beyond, 80
  modeling spaces, 29
  nature of, 8–9
  quality control. *See* testing
  quality judged only by interface, 9
  size, measurement of, 283–284
Software Engineering Institute, 138, 139
software engineering process. *See also* software process
  buying off-the-shelf, 237–238
  creating, 237–238
  described, 237–241
  and process engineer, 263
software process. *See also* quality software process; software engineering process
  deployment, 249–252
  development, 165
  distinct from quality software process, 162–163
  enacting, 253–256
  in large projects, 68
  maturity. *See* process maturity
  as part of quality software process, 162
  process-component. *See* process component, for individual components
  quality of, 19
  and the quality software process, 22–23
software quality
  assuring, 10–11
  defining, 4–7
  elusiveness of, 4

software quality (*cont.*)
nature of software, 8–9
and objective effort, 7–8
solution space. *See also* model of the solution space; team organization, roles in solution space
activities, 32
defined, 31–32
management activities, 31
organizing the roles, 89, 90
specification, and UML, 40–41
spiral-based SDLC, 241–242
sponsor
budget responsibility, 97
impact on project success, 97
and modeling, 28
role in business evaluation, 166
role in prototyping, 198
role on team, 97
and testing results, 341
and training in IIP approach, 252
staff, and reusability, 111–112
staff, and roles. *See* project organizational plan
staff experience, 103, 319
standards
application, 54, 133–136
database, 135
as deliverable, 138, 221
deployment of, 137–138
industrial, 137
length of, 135, 137
levels, 134
modeling, 133–134
not perfect, but followed, 137
organizational, 136–137
process, 134–135
programming language. *See* programming language, standards
project, 136
purpose of, 137
quality, 135–136
quality aspects, 133
and reuse, 53–54
steering committee
impact on quality, 96
membership, 96
project manager, 96
quality manager, 97
role in business evaluation, 166
role on team, 96
and testing results, 341
strategy, testing. *See* testing, planning
stress testing. *See* performance testing
structuring time, 117
sufficiency, 51, 143
Swiss army knife, 119
synchronous teams, 106–107, 116
syntax, 44–46
system architect
influence on reuse, 94
and quality topics, xxviii–xxix
role in
enterprise architecture, 205
package evaluation, 94
persistence design, 190
reuse, 218
system architecture, 208
role, on team, 94
system architecture
as deliverable, 206, 209
system architecture process-component. *See* process-component for, system architecture
system design process-component. *See* process-component for, system design
system designer
desired experience, 90
as interface designer, 91
and quality topics, xxviii–xxix
role in
implementation, 192
reuse, 218
system architecture, 208
system design, 186
role, on team, 90–91
system functionality, 316

system operation, 317
system test, 316

T
TA. *See* transactional analysis
Tao, 5
tape recording. of interviews, 130
task element in a process, 159. *See also* activities and tasks
task plan. *See* project task plan
team organization
  flattening, 109–111
  importance of, 83–84
  and reusability, 111
  roles, common, 95–97
  roles in
    background space, 95
    problem space, 85–89
    solution space, 91–93
  roles in background space, 94
  roles on quality team, 97–100
teams
  avoiding mediocrity, 109
  best-fit approach to creation, 107
  creating, 107–109
  models, 106–107
technological aspects
  of baking example, 20
  measuring, 269
  of a process, 152, 153
telecommuting, 105–106
telephone, 104, 130
test cases
  defined, 328
  as deliverable, 228
  description, 328
  designing, 328–329
  format and content, 329–330
  in Lucky Insurance project, 330–333
  modifying, 334
  verifying, 334
test data
  augmented by previous results, 335
  creating, 312
  as deliverable, 228
  risks of unavailability, 320
test design
  as deliverable, 227
  description, 325
  different from test plan, 342
  format and content, 327–328
  sources of, 325–327
test harness. *See* test script
test manager, testing role, 321
test plan
  as deliverable, 221, 227
  described, 145–146
  different from test design, 342
  in planning process, 319
  and project plan, 145
  reusability aspects, 324
test scripts, 314, 316
test suites, execution of, 336–337
tester
  and quality topics, xxviii–xxix
  role in quality control, 227
  role on team, 93, 100
  testing role, 321
testing
  analyzing risks, 319–321
  approaches
    boundary value, 307, 313
    cyclic, 323
    horizontal testing, 307, 311
    introduced, 306–307
    manual testing, 306, 309–310
    need for, 342
    on non-executables, 308
    partitioning, 307, 312–313
    slicing, 307, 311
    visibility, 307
  architecture
    acceptance test, 316–317
    component test, 315–316
    described, 313–314
    operational testing, 317
    performance testing, 318
    regression testing, 317
    scalability testing, 318
    security testing, 318
    system testing, 316
    unit testing, 314–315

testing (*cont.*)
automated testing, 306
in context, 304–306
cycles, 323–324
database. *See* database, for test results
described, 303
in development environment, 322
environment
common requirements, 322
creating, 322
as deliverable, 228
physical, 320
execution
acceptance criteria, 335–336
analyzing results, 338–340
incident reports, 337
preparation, 334–335
recording test results, 338
reporting, 341
software incidents, 337–338
horizontal, 311
incidents, defined, 337
introduction, 304–306
limiting scope, 306
metrics, 287
organization of, 326
planning, 318–324
resources for, 321
results. *See also* software, incidents
categories, 337
as deliverable, 229
importance of saving, 316
recording and analyzing, 337–341
reporting, 341
reusability in, 324
risks, 319–321
schedules, 322–323
scope of, 319–321
as software quality control, 304–306
status of, 341
strategy. *See* testing, planning
test cases. *See* test cases
test design. *See* test design
test execution, 334–337
test plan. *See* test plan
test planning
analyzing risks, 319–321
described, 318–319
development environment, 322
reusability, 324
test cycles, 323–324
test environment, 321, 322
test plan, 319
test resources, 321
test schedules, 322–323
theory and practice of, 303
time estimates, 322–323
timelines, 323
tools, 310
unit, 314–315
vertical, 311
when to start, 323
when to stop, 323, 338, 342
Thomas, D., 285
time
nonlinear, 12, 102, 281–282
as quality pressure, 12
TOOLS 2000, 28
training
materials, as deliverable, 216
and mentoring, 252
on object-oriented concepts, and testing risks, 319–320
training manager, role in training, 215
training plan, as deliverable, 216
training process-component. *See* process-component for, training
transactional analysis
games in a project, 117–118, 131
history of, 113–114
life positions, 116–117
managing multiple teams, 113
Parent, Adult, and Child ego states, 114–115
in software projects, 113–114
time and effort, 147
triangle, quality context, 15
Trueblood, Elton, 267

U
UML
  advantages, 34
  artifacts, measurement of, 286–287
  common misunderstandings, 34
  and communication, 34
  diagrams
    measurement of, 286–287
    and modeling spaces, 42–44
    in practice, 43
    and respective models, 44
  and documentation quality, 42
  history of, 34–36
  metamodels, 36–38, 156
  models, measurement of, 286–287
  and quality
    of construction, 41
    of documentation, 42
    and processes, 36
    of specification, 40–41
    of visualization, 39–40
  separating from the process, 251
  skills, need in testing, 320
  in teaching object-orientation, 62–63
  as a visual modeling language, 40
UML 2001, xxix
UML-based projects. *See also* projects
  size, 63–69
  types
    data warehouse/conversion, 62
    development, 58
    educational, 62–63
    integration (with legacy), 58–60
    outsourcing, 61–62
    package implementation, 60–61
    typical, 56–58
unit test, 314–315
unplanned, planning for, 283
upgrade/release strategy, as deliverable, 202
usability
  defined, 54
  and domain expert, 89
  great law of, 54–55
  inspections, 124–125
  lesser law of, 55
  principles of, 55
  problem space prototype, 89
  studies, 73
  of systems, and user input, 73
  testing, 342
use cases
  diagrams, measurement of, 286
  documentation, 308
  testing vs. class testing, 327
Use It or Lose It game, 119
user
  in acceptance testing, 316
  communication abilities, 86–87
  and core requirements, 86
  as distinct from end user, 87
  importance to quality, 100
  and model semantics, 48
  in reviews, 126
  role, as acceptance tester, 87
  role in
    deployment, 212
    interface modeling, 182
    project management, 170
    prototyping, 198
    providing quality, 86
    quality control, 227
    quality management, 220
    training, 215
  role, on team, 86, 100
  training in IIP approach, 252
  UML-literacy, 87
user domain, acceptance test, 316
user, end. *See* end user
user interface modeler, role in interface modeling, 182
user representative. *See* user
utopian teams. *See* synchronous teams

V
version control plan, as deliverable, 203
vertical testing, 307, 311
videos, as communication mechanism, 105
visibility testing approach, 307
visual models, 40

visualization, and UML, 39–40
volume testing. *See* performance testing

W
walkthroughs
  described, 123–124
  of quality models, 47
  use of checklists, 123–124
  in white box testing, 308
wall of project area, as communication mechanism, 105
waterfall-based SDLC, 241
website, as communication mechanism, 104
weighting factors, 278, 285–286
"what" of a process. *See* technological aspects
white box testing, 306, 308–309
white cloud, 71
whiteboard, use in workshops, 131
"who" of a process. *See* sociological aspects
WIPRO, 141
Wohlin, C., 268
work ethics, and telecommuting, 106
workshops
  consolidation at end, 133
  controlling conversations, 132
  described, 131–133
  documentation of results, 133
  equipment, 131
  facilitator, 131
  planning for, 131
  play acting, 132
  on quality topics, xxx
  for reviews, 125
  role in inspections, 124
  as source of gaming, 131
  technical vs. facilitator skills, 131
  value of confusion in, 132

X
XP, as off-the-shelf software, 23

Y
Y2K cleanup, 42, 59
Yourdon, Ed, 235