Index

A
abstract class
definition, 9
abstraction
definition, 6
element, 6
accessor methods, 229
advantages, 236
definition, 230
lazy initialization, 234
naming convention, 234
on models, 390
testing, 389
tips, 258
acting out scenarios, 23
actor classes
and external entities, 71
actors
definition, 58
aggregation
applicability, 32
class, 39
defining, 32
definition, 11
deleting objects, 323

example, 11
implementing, 322
retrieving objects, 323
simplifying hierarchies, 152
vs. instance relationships, 322
writing objects, 323
alpha testing, 409
definition, 409
Ambler notation
best of breed, 55
moving to UML, 34
summary, 41
analysis
metrics, 179
testing, 369
analysis-and-design
diagramming, 80
analysis classes, 88
analysis errors, 370
definition, 21, 371
analysis patterns
business entity, 128
contact point, 128
definition, 118
Item-Item description, 126
analysis patterns (continued)
   Place, 131
   Shipping/Billing, 130
analysis testing, 369
   and interface-flow diagrams, 76
   prototype walkthroughs, 370
   use case scenario testing, 370
API. See Application programming interface
applets, 160, 203
   advantages, 160
   and electronic commerce, 214
   definition, 204
application programming interface
   C-APIS, 349
   definition, 350
application servers, 146
   definition, 148
application testing, 403
applications
   no more, 278
   vs. desktops, 279
architecture
   class type, 87
   distributed classes, 148
   distributed objects, 161
   for electronic commerce, 209
   object-oriented C/S, 149
   peer-to-peer, 149
   three-tier C/S, 146
   two-tier C/S, 144
associations. See Relationships
associative tables, 325
   definition, 294
attitude, 222
attributes
   and states, 64
   class, 33
   definition, 6
   effective, 226
   finding, 31
   instance, 33
   metrics, 181, 184
   naming conventions, 226
   non-key, 304
   tips, 257

B
bank case study, 42
BDEs. See Business domain experts
behavioral redundancy, 307
beta testing, 409
   definition, 409
binding, 198
black-box testing, 383
   definition, 384
booleans
   getters, 232
   boundary-value testing
      definition, 385
   boundary values
      and states, 66
   brainstorming, 14
      pointers, 17
built-in test cases, 399
business classes, 88
   and applets, 160
   and CORBA, 356
   and peer-to-peer wrapping, 356
   and persistence, 101
buying, 108
   definition, 89, 101
   identifying, 106
   metrics, 174, 193
   skill requirements, 256
business domain classes, 88
business domain experts (BDEs), 14
business entity pattern, 128
   and contact point, 129
   and Shipping/Billing, 130
business objects
   myths of, 108
business rules
   and testing, 385
   documentation, 239
C
C++
   accessors, 229
   adoption of, 205
   features, 202
   hybrid language, 202
   learnability, 203, 206
Index

- maintainability, 203
- memory leaks, 398
- method size, 185
- metrics, 176, 185
- naming attributes, 228
- simple code, 203
- callback methods, 253
  - definition, 96, 254
- C-APIs, 348
  - skills, 356
- cardinality
  - definition, 11
  - notation, 40
- CASE, 54, 101
  - object-oriented, 216
- centralized mainframes, 141
- checklists, technical reviews, 376
- class
  - contracts, 151
  - definition, 15
  - messages, 38
  - class aggregation
    - definition, 39
  - class attributes
    - definition, 35
    - implications, 37
  - inheritance, 37
  - metrics, 184
  - notation, 34
  - class diagrams
    - Ambler vs. UML, 69
    - and collaboration diagrams, 38
    - and CRC modeling, 26
    - and data models, 79
    - definition, 25
    - metrics, 179
    - notation, 56
    - steps, 25
  - class hierarchy, 9
  - class-integration testing, 396
    - definition, 396
  - class libraries
    - definition, 217
    - reuse, 216
  - class methods
    - definition, 36
- implications, 37
- inheritance, 37
- notation, 36
- class normalization, 303
  - and cohesion, 305
  - and patterns, 127
  - definition, 310
- class relationships
  - definition, 39
- class testing, 392
  - and state diagrams, 393
  - definition, 393
- class-type architecture
  - advantages, 114
  - and two-tier C/S, 145
  - and use cases, 97
  - and wrapping, 356
  - construction, 255
  - definition, 86
  - development implications, 110
  - disadvantages, 114
  - five-layer, 97
  - four-layer, 93
  - project-management implications, 109
  - two-layer, 89
- class types
  - metrics, 193
- classes
  - abstract, 9
  - and class types, 86
  - application-specific, 180, 182
  - as global variables, 231
  - business, 88
  - classification, 106
  - client, 154
  - client/server, 154
  - concrete, 9
  - definition, 5
  - finding, 18, 30
  - getters, 231
  - interface, 75, 88, 98
  - leaf class, 182
  - maintainability, 247
  - naming conventions, 247
  - persistence, 91, 102
Index

460

classes (continued)
  reuse, 247
  root, 9
  root class, 182
  router, 159
  server, 154
  subclasses, 9
  superclasses, 9
  system, 94, 105
  testing, 392
  tips, 258
  wrappers, 344
classes per developer metric, 175
clear-box testing, 384
client classes
  definition, 155
  identifying, 156
client/server
  and object-orientation, 149
  definition, 142
  fat client, 144
  features, 143
  thin client, 144
  three-tier, 146
two-tier, 144
client/server classes
  definition, 155
  identifying, 156
clients
  definition, 142
  features, 143
code
  documentation, 239
  paragraphing, 241
code quality
  methods, 237
  metrics, 185–186, 190
code reviews, 382
  vs. code testing, 382
code testing, 380
  black box, 383
  boundary value, 385
class-integration testing, 396
class testing, 392
clear box, 384
comparison of techniques, 402
coverage testing, 386
inheritance-regression testing, 394
method testing, 389
new techniques, 388
path testing, 386
process, 401
reviews, 382
traditional methods, 383
vs. code reviews, 382
white box, 384
cohesion
  and class normalization, 305
  and subsystems, 154
  definition, 12
collaboration
  defining, 31
  definition, 12
  example, 12
collaboration diagrams, 38, 50
  definition, 69
drawing, 69
  when to draw, 70
collaborator
  definition, 18
definition, 16
color usage, 275
comments
  and maintainability, 239
  metrics, 186–187, 190
component
  assemblers, 109
  builders, 109
  definition, 218
  reuse, 216
cricket class, 9
concurrency
  CORBA, 164
definition, 293
conferences, 421
consensus, 378
consistency, 267
and mental models, 270
  and naming conventions, 226
  and patterns, 134
Index

| D |

- data dictionaries
  - and mapping objects, 330
  - contents, 331
  - definition, 295
- data entity
  - definitions, 77
- data flow, 71
- data flow diagrams, 71
- data justification, 276
- data models, 50
  - and class diagrams, 79
  - definition, 77, 299
  - drawing, 78
  - when to draw, 79
- data normalization, 300
  - and coupling, 316
  - definition, 304
  - ease of data access, 316
  - speed of data access, 316
- data store, 71
- data warehouse
  - definition, 321
- databases
  - data models, 79
- database proxies
  - definition, 293
- debugging
  - wrapper classes, 350
- deliverables, 50
- design
  - and message flow, 92
  - and testing, 379, 398
  - intelligent error messages, 112
  - metrics, 180
  - of interface objects, 281
  - technical reviews, 372
- cost/benefit analysis, 30
- costs of testing, 365
- coupling, 199
  - and accessors, 236
  - and normalization, 316
  - and wrapping, 348
  - between class types, 89
  - definition, 12
- coverage testing, 386
  - definition, 388
- cowboy programmers, 223
- CRC cards
  - definition, 15
  - moving, 19
- CRC modeling, 14
  - advantages/disadvantages, 20
  - and class diagrams, 26
  - and metrics, 174
  - and use cases, 22
  - definition, 15
- CORBA, 163
  - and persistence, 165
  - and security, 166
  - concurrency, 164
  - definition, 167, 355
- IDL, 163, 354
- services, 163
- skills, 356
- wrapping, 354
- context-sensitive menus, 277
- contracts
  - class, 151
  - definition, 152
  - subsystem, 158
- contact point pattern, 128
  - and Shipping/Billing, 130
- constructor, 36
  - definition, 235
- construction
  - and class-types, 255
  - metrics, 185
- constructors, 36
  - definition, 235
- constants
  - getters, 230
- and UI design, 273
- metrics, 179
- steps, 17
- testing, 370
- crossing lines, 51
- CRUD, 90
- definition, 293
- cursory reviews, 374

© in this web service Cambridge University Press
www.cambridge.org
Index

[462]
design (continued)
  testing, 372
design issues
  class methods, 36
design patterns
  definition, 118
  Proxy, 120
  Roles Played, 122
  Singleton, 119
  state, 122
design testing, 372
  and sequence diagrams, 63
  requirement-verification matrices, 379
desktop
  vs. applications, 279
desktop metaphors, 271
destructors, 36
development
  incremental, 225
  small steps, 225
development/maintenance
  trade-off, 224
DFD. See Data flow diagrams
diagramming
  avoid cluttering, 53
  avoid crossing lines, 51
  avoid curved lines, 52
  avoid diagonal lines, 52
  bubble size, 53
  how it fits together, 80
  on white boards, 74
  simplify, 51
distributed applications
  history of, 140
  distributed classes, 148
  applets, 160
  definition, 148
  disadvantages, 161
  Java, 161, 215
  vs. distributed objects, 215
distributed databases, 337
distributed objects, 161
  definition, 162
  vs. distributed classes, 215
  distribution plans, 406
do/ (keyword), 68
documentation, 32
  internal to code, 239
  maintainability, 236
  of accessors, 236
  of attributes, 236
  of business rules, 239
  of interface objects, 284
  of methods, 237
  of prototypes, 283
  operations, 407
  standards, 407
tips, 241
  user, 406
drag and drop, 264
DSL. See Dynamic shared libraries
dumb terminals, 141
dynamic-link libraries (DLLs), 280
dynamic models. See State diagrams
dynamic shared libraries, 326, 348
  definition, 328, 350
  skills, 356
  dynamic SQL
  definition, 330
  vs. static, 329
dynamic typing
  definition, 202
E
e-cash. See Electronic cash
e-commerce. See Electronic commerce
  80/20 rule, 189
electronic cash, 211
electronic commerce, 207
  and Java, 208
  architecture, 209
  definition, 210
development checklist, 214
  definition, 210
  international issues, 211
  payment processing, 210
  shipping and customs, 212
taxes, 212
  virtual products, 212
  encapsulation
  and languages, 200
Index

463

and wrapping, 345
definition, 6
enforcement, 199
example, 7
English descriptors, 227
entity-relationship diagrams. See Data models
ER diagrams. See Data models
error handling, 251
estimating projects, 173
throughout the lifecycle, 174
evaluating prototypes, 283
extensibility
and wrapping, 358
definition, 86
external entity
and actor classes, 71
definition, 71
forward engineering, 218
frameworks
definition, 218
reusability, 217
full life-cycle object-oriented testing.
steps of, 368
function points, 177, 180
function testing, 404
and UAT, 404, 408
and use cases, 404
definitions, 404
G
garbage collection, 198
control of, 201
definition, 202
getter methods, 229
definition, 230
documentation, 236
for booleans, 232
for classes, 231
for constants, 230
testing, 390
global usage metric, 183
global variables
and class attributes, 184
and classes, 231
and Singletons, 183
graying out, 276
H
hacking
and evaluating code, 185
vs. incremental development, 223
vs. iterative development, 113, 223
human–computer interaction (HCI)
boundary, 97
human factors, 269
definition, 272
Hungarian notation, 228
hybrid languages
C++, 202
definition, 202
vs. pure, 201
Index

IDL. See Interface definition language

IDL. See Interface definition language

- ignored methods
definition, 184
metric, 184

impedance mismatch, 299, 311
definition, 311
implementation, 33
incremental development, 33, 225
and testing, 369
vs. hacking, 223
index cards, 14
information hiding, 229
definition, 7
example, 7
inheriting
and interface objects, 263
and languages, 200
and mapping objects, 315
and state diagraming, 65
and testing, 394
applicability, 31
class hierarchy, 9
defining, 31
definition, 8
example, 8
implementing in relational DBs, 317
of class attributes, 37
of class methods, 37
metrics, 181–183
multiple, 9, 124
pure, 65, 184
simplifying hierarchies, 152
single, 9
tree depth, 182
inheritance regression testing, 394
definition, 395
test cases, 399
inheritance tree depth metric, 182
initial state
definition, 64
installation testing, 406
definition, 406
instance attributes
definition, 35
instance methods
definition, 36
instance relationships
and data models, 77
definition, 10
example, 10
implementing, 322
vs. aggregation, 322
instances. See Objects
instantiating
definition, 5
integration testing
of classes, 396
vs. class testing, 392
interface classes, 88
and applets, 160
and business logic, 100
and clients, 151
and screen scraping, 356
and use cases, 100
buying, 107
definitions, 75, 89
developing, 112
example, 98
identifying, 106
metrics, 193
skill requirements, 255
supporting multiple language, 211
interface definition language (IDL), 163
definition, 167, 355
interface-flow diagrams
and prototyping, 74
definition, 75
drawing, 75
for OOUIs, 281
when to draw, 76
interface layer
advantages, 99
interface objects
and inheritance, 263
and polymorphism, 263
definition, 75, 264
design, 281
documenting, 284
Index

international commerce, 209
Internet
definition, 210
development, 203
electronic commerce, 207
newsgroups, 421
Intranet, 208
definition, 210
"is-a" relationships, 8
"is-like" relationships, 8
ISO 9000, 410
ISO 9001, 410
definition, 411
ISO 9003, 410
definition, 411
Item-Item description pattern, 126
and contact point, 129
and Place, 132
iterative development, 27
modeling, 80
testing, 415
vs. hacking, 113, 223

J
Java, 203
and electronic commerce, 208, 213
and ObjectCOBOL, 215
and Smalldtalk, 215
applets, 160, 203, 280
distributed classes, 215
features, 203
future of, 204, 215
naming attributes, 228
sample code, 204
virtual memory, 204

K
key
definitions, 77, 294
foreign, 301
vs. object identifiers, 312
keystrokes, simulating, 352

L
languages
attribute naming conventions, 227
comparison of, 207
features, 198
hybrid, 199
pure vs. hybrid, 201
purity, 199
Law of Demeter, 247
layer
definition, 86
lazy initialization, 234, 390
definition, 235
learning curve, 420
legacy applications
and electronic commerce, 216
definition, 87, 351
reuse of, 357
tips, 352
wrapping, 347, 351
legacy code
wrapping, 345, 348
lines of code
metric, 188
LOC. See lines of code
locks
definition, 293

M
maintainability
and ancestors, 229
and C++, 203
and comments, 239
and paragraphing, 241
and parenthesis, 245
definition, 86
factors, 224
of attributes, 229
of classes, 247
of documentation, 236
of methods, 244
30-second rule, 245
maintenance
and wrapping, 357
Index

maintenance (continued)
costs, 224
many-to-many relationships, 250
many-to-one relationships, 250
mapping objects
and data dictionaries, 330
and inheritance, 315
seven commandments, 332
to flat files, 296
to relational databases, 326
master test plan, 364, 413
definition, 414
member function. See Methods
memory leaks, 200, 398
definition, 202
memory management, 198
memory management
and C-APIS, 349
mental models, 269
definition, 272
mentoring, 420
mentors, 421
menus, pop-up, 277
message dispatcher, 253
definition, 96, 255
message flow
between class types, 94
restricting, 89, 92
message-invocation box
definition, 60
messages
definition, 12
dispatcher, 96, 253
Law of Demeter, 247
metrics, 186–187
testing, 397
to classes, 38
messaging
definition, 12
metaphors, 271
definition, 272
desktop, 271
method count metric, 180
method response metric, 186
method size metric, 185
method testing, 389
definition, 389
tips, 392
methodology, 54
methods
accessors, 229
and transitions, 65
callback, 96, 253
class, 35
comments, 186
definition, 6
documentation, 237
finding, 31
getters, 229
ignored, 184
instance, 35
maintainability, 244
metrics, 180, 184–187, 190
naming conventions, 237
overriding, 184
quality, 237
refactoring, 243
setters, 229
should do one thing, 243
should do something, 242
size, 185
testing, 389
tips, 257
metrics
and C++, 176
and class types, 193
and Smalltalk, 176
automation of, 192
classes per developer, 175
collection of, 192
comments per method, 186, 190
definition, 172
feature points, 177
for analysis, 179
for business classes, 193
for C++, 185
for construction, 185
for design, 180
for estimating projects, 173
for improving development, 178
Index

467

for improving your development approach, 191
for interface classes, 193
for persistence classes, 193
for Smalltalk, 185, 187
for source code, 185
for system classes, 193
for testing, 189
for tool selection, 190
function points, 177, 180
global usage, 183
inheritance tree depth, 182
lines of code, 188
method count, 180
method response, 186
method size, 185
number of children, 183
number of class attributes, 184
number of ignored methods, 184
number of instance attributes, 181
number of key classes, 174
number of reused classes, 176
number of use cases, 179
percentage of commented methods, 187, 190
percentage of key classes, 179
percentage of tested use-case scenarios, 189
person days per class, 174
printout height, 188
problem reports per class, 189
strings of message sends, 187
success factors, 191
use cases, 189
uses, 173
middleware
definition, 142
object request brokers, 161
mindset change, 421
MIS (management information system), 295
multi-platform support, 99
multiple inheritance, 9
and C++, 202
and OODBMS, 337
and Roles Played pattern, 124
simulating, 124

N
naming conventions
attributes, 226
for accessors, 234
for methods, 237
Hungarian notation, 228
nested logicals, 246
networks
features, 143
normalization, 299
data, 300
not-invented-here (NIH) syndrome, 135
notation
class diagramming, 56
comparison, 56
definition, 54
number of children metrics, 183
number of class attributes metric, 184
number of ignored methods metric, 184
number of instance attributes metrics, 181
number of key classes metric, 174
number of reused classes metric, 176
number of use cases metric, 179

O
object adapter
definition, 294
Object Database Management Group
and CORBA, 165
object diagramming. See Collaboration diagrams
object identifiers
and business meaning, 312
and polymorphism, 315
assigning, 316
definition, 121, 292
implementing, 312
strategies, 313
uniqueness, 314
Index

object identifiers (continued)
  vs. keys, 312
object linking and embedding, 93
Object Management Group, 163, 167, 354
definition, 355
object-message diagrams. See Collaboration diagrams
object modeling technique, 55
object models. See Class diagrams
object-orientation
  adoption of, 205
  and client/server, 149
object-oriented client/server, 149
  steps of, 150
object-oriented databases. See OODBMS
object-oriented user interface, 262
  and wrapping, 346
definition, 264
differences, 268
  example, 265
  features of, 262
  modeling, 281
  vs. standard GUI, 268
object/relational databases, 339
definition, 340
object request brokers, 161
  and TP monitors, 161
definition, 162
object streaming, 167
ObjectCOBOL
  adoption of, 205
  and electronic commerce, 213
  and Java, 215
  features, 205
  naming attributes, 228
  sample code, 205
objects
  definition, 5
  interface, 262
  versioning, 337
ODBC. See Open database connectivity
ODMG, 294
OID. See Object identifiers
OLE. See Object linking and embedding
OMG. See Object management group
OMT. See Object modeling technique
one-to-one relationships, 251
OOCRUD, 90, 104, 328
  and CORBA, 164
definition, 293
OODBMS, 334
definition, 335
  features, 335
  myths, 338
  scalability, 338
13 golden rules, 335
OOCS. See Object-oriented client/server
OOSDLC, 27
OOUI. See Object-oriented user interface
open database connectivity (ODBC)
  definition, 328
OpenDoc, 93
operations testing, 407
definition, 407
optimistic locking
definition, 294
optionality
definition, 11
notation, 40
OQL
definition, 91, 294
ORB. See object request brokers override
definition, 9, 184
overload. See Override

P
paragraphing, 241
definition, 242
parenthesis, 245
Pareto's rule. See 80/20 rule
partitioning classes, 113
path testing, 386
definition, 388
Index

patterns
  advantages, 134
  analysis, 118
  and reuse, 134
  as a buzzword, 136
  business entity, 128
  contact point, 128
  definition, 118
  design, 118
  disadvantages, 135
  discovery, 133
  effective use, 132
  item-item description, 126
  list of, 136
  Place, 131
  Proxy, 120
  Roles Played, 122
  Shipping/Billing, 130
  Singleton, 119
  state, 122
  using them together, 129
  payment processing, 210
  peer-to-peer architecture, 149
    definition, 150
  peer-to-peer wrapping, 353
    definition, 354
  skills, 356
  percentage of commented methods metric, 187
  percentage of key classes metric, 179
  percentage of tested use-case scenarios metric, 189
  persistence
    and CORBA, 164
    and data models, 79
    and proxies, 122
    classes, 102
    definition, 9, 292
    persistent memory, 10
    persistent objects, 10
    transitory objects, 10
    persistence classes, 111
    and C-APIs, 356
    and DSLs, 356
  buying, 108
  definition, 91, 102
  development, 111
  identifying, 107
  metrics, 193
  skill requirements, 256
  persistence layer, 90
    advantages, 103
    and OOCRUD, 104
    definition, 292
    implementing, 93
    persistence mechanism
      definition, 91, 292
      flat files, 295
      object-relational databases, 339
      OODBMS, 334
      relational databases, 298, 311, 326
      security, 337
      persistent memory
        definition, 10
      persistent objects
        definition, 10
        example, 10
      person days per class metric, 174
      pessimistic locking
        definition, 293
      pilot testing, 409
        definition, 409
      pinball SDLC, 27, 414
      Place pattern, 131
      polymorphic
        definition, 13
      polymorphism
        and case statements, 246
        and interface objects, 263
        and languages, 200
        and OIDs, 315
        and testing, 397
        definition, 13
        example, 13
      pop-up menus, 277
      portability
        and Java, 203
        definition, 86
      pre/(keyword), 68
      printout height metric, 188
Index

problem reports per class metric, 189
process models
  and DFDs, 72
definition, 71
drawing, 72
tips, 74
  when to draw, 74
productivity
  and patterns, 134
program code
  root of all evil, 107
programming, 32
project
  initiation, 30
teams, 109
project estimating
difficulties, 175
tips, 178
project management
  and class-type architecture, 109
prototype walkthroughs, 370
definition, 371
prototypes
  definition, 371
prototyping, 19, 112
  and interface-flow diagrams, 74
  and testing, 370
definition, 283
documentation, 283
prototyping
OOUIs, 281
steps of, 282
tips and techniques, 284
tools, 286
types of, 286
proxy
database, 293
proxy pattern, 120
pure inheritance, 65
pure languages
definition, 202
  vs. hybrid, 201

QA. See Software quality assurance

R
  read lock
definition, 293
  recursive transition
definition, 64
  refactoring, 243
definition, 244
  regression testing
definition, 367
  inheritance, 394
test cases, 369
relational databases, 298
  reasons for using, 298
relationships
  aggregation, 11
  and CORBA, 166
  and foreign keys, 324
cardinality, 11
class, 39
  finding, 31
implementing, 249, 322
in data models, 77
instance, 10
  "is-a," 8
  "is-like," 8
  "is-part-of," 11
many-to-many, 250, 325
many-to-one, 250
one-to-many, 324
one-to-one, 251, 324
optionality, 11
navigation direction, 41
  "uses," 58
repositories
definition, 218
requirement-verification matrices, 379
  example, 380
responsibility, 15
  finding, 18
reuse
  and inheritance, 183
  and patterns, 134
  and wrapping, 357
class libraries, 216
Index

ISO 9000 says nothing about quality, 411
iterative vs. hacking, 223
lines of code, 188
mapping objects to RDBs, 333
persistance mechanism market, 340
pure vs. hybrid debate, 201
UI standards are a must, 270
when is a language OO, 200
your UI isn’t OO ..., 269
scaffolding, 347
definition, 348
scalability
OODBMS, 338
scheduling
and prototypes, 286
Scott’s Soapboxes
automatic garbage collection, 200
coverage and path testing aren’t as important anymore, 387
don’t hardcore SQL, 327
estimating is tough, 175
future of Java, 204, 215
good C/S implementation, 143
is history repeating itself?, 205
Index

setter methods (continued) testing, 390
Shipping/Billing pattern, 130
single inheritance, 9
Singleton pattern, 119
and globals, 183
skill requirements business classes, 256
interface classes, 255
persistence classes, 256
system classes, 256
skills transfer, 420
Smalltalk, 280
accessors, 229
and electronic commerce, 213
and Java, 215
execution speed, 207
features, 207
for learning C++, 206
method size, 185
metrics, 176, 185, 187
naming attributes, 228
sample code, 207
testing, 392
2NF. See Second normal form
software quality assurance, 410
definition, 411
2ONF. See Second object normal form
SQA. See Software quality assurance
SQA departments, 411
SQL
definition, 91, 294
dynamic, 329
static, 329
standards abbreviations, 227
for UI design, 270, 273
Hungarian notation, 228
state diagrams and testing, 393
definition, 64
drawing, 66
hints, 67
inherence of, 65
when to draw, 68
states and attributes, 64
definition, 64
final, 64
initial, 64
substate, 67
superstate, 67
state-transition diagrams. See State diagrams
static binding
definition, 202
static SQL
definition, 330
vs. dynamic, 329
stress test plan, 405
definition, 406
stress testing, 405
definition, 406
strings of message sends metric, 187
subclasses
definition, 9
substate
definition, 67
subsystem contracts
definition, 158
subsystems
and router classes, 159
and server classes, 158
definition, 155
identifying, 150, 154
implementing, 159
success factors, 191
superclasses
definition, 9
superstate
definition, 67
system classes
and C-APIS, 356
and DSLs, 356
buying, 109
definition, 94, 105
developing, 110
identifying, 107
metrics, 193
skill requirements, 256
system integration, 346
system testing, 403
definition, 403
function testing, 404
installation testing, 406
operations testing, 407
stress testing, 405
tips, 404
volume testing, 405
systems programming, 202

team development, 217
technical-design reviews, 372
advantages, 378
definition, 373
planning, 375
process, 373
what to review, 373
technical review checklists, 376
technical review plans, 375
definition, 376
test cases
and regression testing, 369
built-in, 399
definition, 364
implementing, 398
separate hierarchy, 399
test plans, 401
definition, 364
review of, 401
stress test, 405
UAT, 408
test scripts, 400
testing, 32
alpha testing, 409
analysis testing, 369
and good design, 379, 398
and incremental development, 369
and ISO 9000, 410
and polymorphism, 397
and QA, 410
and state diagrams, 393
application testing, 403
beta testing, 409
black box, 383
boundary value, 385
class-integration testing, 396
class testing, 392
clear box, 384
code testing, 380
costs of, 365
coverage testing, 386
definition, 364
design testing, 63, 372
FLOOR, 364
FLOOR overview, 414
function testing, 404
in the large, 403, 414
in the small, 414
inheritance-regression testing, 394
installation testing, 406
issues in, 21
messages, 397
method testing, 389
metrics, 189
OO vs. structured, 367, 388
operations testing, 407
paradigm, 388
path testing, 386
pilot testing, 409
process, 401
realities of, 365
regression testing, 367
requirement-verification matrices, 379
Smalltalk code, 392
stress testing, 405
subclasses, 400
system testing, 403
test cases, 364
test logs, 364
test object sets, 388
test plans, 364
test scripts, 364
tips, 412
tools, 217
traditional methods, 383
training in, 412
throughout the SDLC, 364
use-case scenario testing, 20, 22
user-acceptance testing, 407
user interface, 76
testing (continued)
  volume testing, 405
  white box, 384
testing in the large, 414
definition, 415
testing in the small, 414
definition, 415
test-procedure scripts
definition, 364
thin-client approach, 144
third normal form, 302
  and 30NF, 307
definition, 304
third object normal form, 307
definition, 311
13 golden rules of OODBMS, 335
30-second rule, 185, 245
three-tier client/server, 146
  advantages, 147
definition, 148
tips
  achieving the benefits of distribution, 148
  analysis patterns, 125
  application should be self-updating, 146
  assign legacy screens unique IDs, 352
  assigning people to classes, 189
  be proactive with beta testers, 410
  combine test cases, 394
  coding standards we input to code reviews, 382
  concentrate on typical situations when testing another's code, 386
diagramming, 52
document interface objects once they're stable, 284
drawing state diagrams, 67
effective documentation, 241
error handling, 253
errors are usually in your new code, 383
establishing your priorities, 190
estimating, 178
  for accessors, 258
  for attributes, 257
  for classes, 258
  for methods, 257
  for metrics, 191
  for OO development, 256
  for system testing, 404
  for 20NF, 306
  get someone who can draw, 281
  get the design right first, 379
  good designs require less testing, 398
  how to remember the rules of 3NF, 303
  inheritance and relational DBs, 316
  lazy initialization and persistence, 235
  look at subsets of behavior, 308
  look to past problems to create checklists, 377
  maintenance of methods, 244
  method testing, 392
  naming conventions, 226
  naming loop counters, 227
  people first, 242
  performing reviews, 377
  placing class types, 141
  process modeling, 74
  prototyping, 284
  put classes into 30NF only when it makes sense, 310
  return intelligent errors, 112
  reuse existing classes, 247
  set naming conventions for methods, 13
  stabilize your code before testing it, 381
  test for inheritance, 396
testing, 412
timestamps vs. datetimestamps, 123
use a spreadsheet for requirement-verification matrices, 380
Index

use data warehouses for reporting, 321
user interface design, 273, 287
when to use the Roles Played pattern, 125
your prototype must be doable, 283
3NF. See Third normal form
3ONE. See Third object normal form
tool selection
metrics, 190
training and education
C++, 206
ObjectCOBOL, 206
training courses, 420
training plans, 406
transaction
definition, 293
transaction processing (TP) monitors, 161
transient objects. See Transitory objects
transitions
and methods, 65
definition, 64
recursive, 64
transitory objects
definition, 10
example, 10
two-tier client/server
definition, 144
typing, 198

definition, 58
drawing, 58
example, 57
when to use, 59
use-case scenario testing, 370
definition, 371
use cases
and class-type architecture, 97
and function testing, 404
and interface classes, 100
and sequence diagrams, 60
and user acceptance test planning, 23
creating, 30
defining, 19
definition, 20, 25, 58
metrics, 179, 189
reuse, 59
scenario testing, 20, 22, 25
scenarios, 20–21, 23
user acceptance test plan
and use cases, 23
user-acceptance testing, 407
and function testing, 404, 408
definition, 408
user interface design
and metaphors, 271
and standards, 270
color usage, 275
consistency, 267
field alignment, 276
fonts, 275
grouping widgets, 277
screen density, 277
screen navigation, 274
testing, 370
tips, 273, 287
wording messages, 274
user interfaces
and database access, 100
and interface-flow diagrams, 74
implementing, 98
user-interface specialists, 99
user requirement reviews
definition, 371

U

UAT. See User acceptance testing
UML. See Unified modeling language
unified modeling language, 34
definition, 35
simplified class diagramming notation, 42
unit testing
definition, 389
vs. class testing, 392
vs. method testing, 389
use-case diagrams
Index

user requirements
reviews, 371
users
and testing, 366
working with, 226
V
versioning, 337
virtual machine, 204
definition, 204
virtual products, 212
volume testing, 405
definition, 406
W
walkthroughs
prototype, 370
whitespace, 239
definition, 240
whiteboards, 74
white-box testing, 384
definition, 385
widgets, 274
grouping, 277
World Wide Web (WWW), 203
definition, 210
wrapper class, 344
definition, 345
wrappers, 344
database, 328
definition, 345
wrapping, 344
advantages, 357
and coupling, 348
and electronic commerce, 216
and extensibility, 358
and maintenance, 357
wrapping
and system integration, 346
approaches, 346
C-APIs, 348
comparisons, 355
CORBA, 354
database access, 91
debugging issues, 350
definition, 91
disadvantages, 357
don’t, 347
dynamic shared libraries, 348
hardware, 347
legacy applications, 347, 351
legacy code, 345
operating systems, 347
peer-to-peer, 353
reasons for, 344
screen scraping, 351
skills, 356
technologies, 348
the system layer, 94
write lock
definition, 293
WWW.AmbySoft.Com, 421
WWW. See World Wide Web