

Index

- active clause coverage (ACC)
 - ambiguity, 108–109
 - definition, 108
- activity diagram, 89–90
- All-Coupling-Defs-Uses (ACDU), 255
- All-Coupling-Sequences (ACS), 255
- All-Poly-Classes (APC), 255
- All-Poly-Coupling-Defs-Uses (APCDU), 256
- ANSI/IEEE Standard, 225
- architectural design, 6
- Ariane rocket, 8
- automatic test data generation, 289–290

- basic block, 28, 52–55, 178, 186, 259, 270, 278
 - definition, 52
- best effort touring, *see* tour, best effort
- black-box testing
 - definition, 21
- block, 150, 153
- bottom-up testing
 - definition, 22
- bypass testing, 258

- characteristic, 151, 153
 - examples, 151, 152, 155
 - functionality-based, 153–155, 158
 - interface-based, 153–157
- CITO, *see* class integration test order
- class integration test order, 218–219, 222, 237
 - definition, 219
- clause
 - definition, 105
- combination strategy, 160
- component, 6, 11, 65, 73, 191–192, 217–218, 236–237, 256–260
 - definition, 217
- concurrency, 268
- connector, 6
- control flow graph, 259
- controllability, 120, 263, 284–286
 - definition, 14
- coupling path, 247–256
- coupling sequence, 250–256
- coverage analysis tool, 269
 - definition, 268
- coverage criterion
 - definition, 17
- coverage level
 - definition, 18
- criteria
 - ACoC, 160, 163
 - ADC, 48, 50, 70, 79, 102
 - ADUPC, 48–51, 70, 102, 103
 - AUC, 48–51, 70, 79, 102, 271–272
 - BCC, 162–163
 - CACC, 109–111, 113, 115, 118–119, 126–129, 133, 134, 142, 147–148, 188, 272
 - CC, 106, 113, 116–117, 126, 132, 187
 - CoC, 107, 113, 117
 - CPC, 36, 50
 - CRTC, 36, 50
 - CUTPNFP, 113, 142–149
 - DC, 172
 - EC, 34, 50–51, 54, 65–66, 79, 90, 96, 101, 172, 187, 268–271
 - ECC, 160–161, 163
 - EPC, 35, 50, 79
 - GACC, 109, 113, 117–119, 134, 142, 147, 148, 188
 - GICC, 112
 - IC, 113, 139–141, 149
 - MBCC, 162–163
 - MC, 175
 - MOC, 175
 - MPC, 175
 - NC, 33–35, 50, 54, 65–67, 78, 87, 90, 96, 101, 172, 186, 268–271
 - PC, 106–109, 112, 113, 115, 116, 118–120, 125, 128–129, 132, 134, 140
 - PDC, 172
 - PPC, 35–36, 39, 50–51
 - PWC, 161, 163
 - RACC, 110–111, 113, 115, 118–119, 134, 142, 144, 146–148, 188

320 Index

- criteria (*cont.*)
 - RICC, 112, 113
 - SMC, 178
 - SPC, 36, 90, 96
 - SRTC, 36, 50
 - TSC, 172
 - TWC, 161–163
 - UTPC, 113, 140–146
 - WMC, 179
- criteria subsumption, *see* subsumption
- criterion
 - definition, 17
 - explanation, 17–18
- debugging
 - definition, 13
- def, 44–45
 - coupling-def, 247
 - first-use, 247
 - last-def, 72, 247
 - definition, 68
 - example, 69–70
- def-clear, 45, 247
- def-pair set, 46
- def-path set, 45
- deploy, 287
- detailed design, 6, 223
- determination
 - definition, 108
 - examples, 108–111, 116–119, 126–127, 132, 134–146
 - explanation, 107, 113–115
- detour, 39
 - definition, 38
 - explanation, 38–39
- Disjunctive Normal Form (DNF), 138
- driver, 218
 - definition, 218
- du-pair, 44, 57–59, 69–70
 - interprocedural, 72–73
- du-path, 45–49, 51, 57–59, 69–70
- dynamic testing
 - definition, 22
- economics, 286–287
- edge, 27–30
- embedded, 262–265, 287–288
- emergent, 280–283
- error
 - definition, 12
 - examples, 7–8, 12–13
 - explanation, 12
 - timeliness, 264
- example Java
 - cal(), 132, 190
 - checkIt(), 130
 - countPositive(), 16
 - findLast(), 16
 - findVal(), 189
 - lastZero(), 16
 - numZero(), 12
 - oddOrPos(), 16
 - power(), 190
 - Quadratic, 71
 - Queue, 85
 - Stutter, 79
 - takeOut(), 74
 - TestPat, 56
 - trash(), 74
 - TriTyp, 121, 152–163
 - twoPred(), 130
 - union(), 10
- executable test script
 - definition, 15
- exit commands
 - definition, 15
- expected results
 - definition, 14
- failure
 - definition, 12
 - examples, 7–8, 12–13
 - explanation, 12
 - timeliness, 264
- fault, 142, 288
 - definition, 12
 - examples, 7–8, 12–13
 - explanation, 12
 - specific types, 142
 - timeliness, 263
- first-use, 27, *see* use, first-use
- generator, 19, 171
 - definition, 18
- genetic algorithms, 290
- Goldilocks problem, 216
- grammar, 170–172
 - ground string, 173
 - nonterminal, 171
 - production, 171
 - rule, 171
 - start symbol, 171
 - terminal, 171
- graph, 27–52
 - case structure, 54
 - double-diamond, 30–91
 - for structure, 54
 - if structure, 53
 - if-else structure, 52
 - SESE, 30
 - while structure, 53
- graph coverage
 - definition, 33
- Graphical User Interface, 260–262
- implementation, 6
- implicant, 138
 - prime, 140
 - redundant, 140
- inactive clause coverage (ICC)
 - ambiguity, 111–112
 - definition, 111
 - examples, 112
- infeasible, 36, 39, 59, 77, 103, 112–113, 165
 - CACC and RACC, 110
 - subsumption, 20
 - test requirements, 18
- infection, 13, 178, 284–285
- input
 - invalid, 173, 261, 283

- usage distribution, 285
- valid, 173
- input domain, 150, 152
- model, 151, 152
- Input Domain Model (IDM), 152–158
 - constraints, 165
- instrumentation, 268–272
 - data flow coverage, 271–272
 - definition, 268
 - edge coverage, 270–271
 - logic coverage, 272
 - node coverage, 268–270
- intermediate design, 222
- Java bytecode, 270, 276–277
- Java reflection, 270, 276–277
- jelly bean, 17–20
- Karnaugh maps, 140, 144–146
- last-def, *see* def, last-def
- literal, 138
- maintenance changes, 216
 - adaptive, 216
 - corrective, 216
 - perfective, 216
 - preventive, 216
- major clause, 107–115, 126–127
 - definition, 107
- Mars lander, 8
- minor clause, 107–115, 126–127
 - definition, 107
- misuse case, 283
- mock
 - definition, 218
- MSG, *see* mutation, schema-based
- muJava, 277
- mutation
 - adequacy, 181
 - bytecode translation, 277
 - dead, 177, 180
 - effective operators, 182
 - equivalent, 177
 - interpretation, 274
 - java reflection, 276–277
 - kill, 175, 177
 - mutant, 173
 - operator, 173, 182–185
 - real-time, 265
 - schema-based, 274–277
 - score, 175
 - selective, 182
 - separate compilation, 274–275
 - SMV, 198–201
 - specification, 198–201
 - stillborn, 177
 - strong, 178–180
 - strongly kill, 178
 - tool building, 272–277
 - trivial, 177
 - weak, 178–180, 186
 - weakly killing, 179
 - XML, 203–204
- near false point, 142
- node, 27–30
 - final, 27
 - initial, 27
- node coverage
 - definition, 33, 34
- object-oriented
 - class, 236
 - data abstraction, 236
 - data flow testing, 247–256
 - dynamic binding, 236
 - inheritance, 236
 - inheritance fault, 240–247
 - inter-class testing, 237
 - inter-method testing, 237
 - intra-class testing, 237
 - intra-method testing, 237
 - override, 238
 - polymorphism, 236, 238
 - subclass inheritance, 236
 - substitution principle, 236
 - subtype inheritance, 236
 - testing, 236–256
 - yo-yo graph, 239–240
- observability, 263, 284–286
 - definition, 14
- oracle, 230–231
 - consistency check, 231–232
 - data redundancy, 232–233
 - definition, 230
 - direct verification, 230
 - redundant computation, 231
- partition, 150, 153
 - complete, 151–152, 155, 158
 - disjoint, 150–152, 155, 158
- path, 29–32
 - du-path, *see* du-path
 - prime, 51
 - definition, 35
 - deriving, 39–42
 - examples, 37
 - simple, 35–36, 39, 51
 - test-path, 30
 - test-path, 30
- path expressions, 91, 93–96, 99
- path product, 91
- Pentium, 7
- postfix values
 - definition, 15
- predicate, 106
 - definition, 104
 - examples, 105
 - minimal, 140
- prefix values
 - definition, 15
- prime path, *see* path, prime
- propagation, 13, 178, 284–285
- proper subterm, 140
- quality assurance, 225–226
- reachability, 13, 178, 284–285
- real-time, 262–265, 268
 - environment, 262

322 Index

- real-time (*cont.*)
 - reactive, 262
 - response time, 263
 - tasks, 262
 - timeliness, 262, 264
- recognizer, 19, 171
 - definition, 18
- regression test
 - inclusive, 217
 - modification-revealing, 217
 - precise, 217
- regular expression, 170–171, 201–202
- regular expressions, 91, 94–96
- reproducibility, 263
- requirements analysis, 4–221
- RIP model, 13, 178, 284
- round trip, 36

- safety, 280–284, 287
- safety constraints, 135
- security, 204, 205, 235, 257, 280–284, 288
- SESE graph, *see* graph, SESE
- sidetrip, 39, 103
 - definition, 38
 - explanation, 38–39
- simple path, *see* path, simple
- static testing
 - definition, 22
- stub, 218
 - definition, 218
- subgraph, 27
- subpath, 29
- subsumption, 19, 21, 23, 34, 50, 112–113, 142–144, 163
 - ACC, IC and UTPC, 141–142
 - clause and predicate coverage, 106–107
 - definition, 19
 - explanation, 19–20, 289
 - graph, 142
 - infeasible, 20
 - mutation, 186–189
 - partitioning criteria, 163
 - predicate coverage and ACC, 109, 129
 - predicate coverage and IC, 140
- subsystem design, 6
- system and software design, 221–222

- term, 138
- test action, 220–223
- test case
 - definition, 15
- test case values
 - definition, 14
- test design, 220–223
- test driver, *see* driver
- test failure
 - definition, 13
- test influence, 220–223
- test path, *see* path, test-path
- test plan, 225–230
 - mission, 226
 - strategic, 226
 - tactical, 226
- test process, 219–220
- test requirement
 - definition, 17
 - explanation, 17–18
- test set
 - definition, 15
- test stub, *see* stub
- testability, 222, 284–286
- testing
 - acceptance, 6
 - definition, 13
 - deployment, 220, 224
 - integration, 6, 217–218, 220, 222–224, 288
 - definition, 217
 - module, 6, 218, 288
 - regression, 215–217, 224–225, 231
 - definition, 215
 - system, 6, 222, 288
 - unit, 6, 218, 222–224, 288
- top-down testing
 - definition, 22
- tour, 31, 32, 39, 70
 - best effort
 - definition, 39
 - definition, 38
 - explanation, 38–39
- Traffic Collision and Avoidance System (TCAS), 232–233
- transaction flow graph, 89

- unique true point, 140
- usability testing, 260
- use, 44–45
 - coupling, 247
 - first-use, 72
 - definition, 69
 - example, 69–70
- use case, 87–90

- validation
 - definition, 11
- verification
 - definition, 11
- verification values
 - definition, 15
- version control, 215, 225
- visit, 31, 32

- web, 287
 - client-side, 257–258
 - dynamic page, 256
 - server-side, 258–259
 - site, 256
 - static page, 256
 - static site, 256
 - web application, 256–259, 287
 - test case, 257
 - web service, 256–260
- white-box testing
 - definition, 21