

# Contents

|                                                                |               |
|----------------------------------------------------------------|---------------|
| Preface . . . . .                                              | 15            |
| Acknowledgements . . . . .                                     | 22            |
| <br><b>Part I: Preliminaries</b>                               | <br><b>24</b> |
| <b>1. Basics of Software Testing</b>                           | <b>25</b>     |
| 1.1. Humans, errors, and testing . . . . .                     | 25            |
| 1.1.1. Errors, faults, and failures . . . . .                  | 27            |
| 1.1.2. Test automation . . . . .                               | 28            |
| 1.1.3. Developer and tester as two roles . . . . .             | 28            |
| 1.2. Software quality . . . . .                                | 29            |
| 1.2.1. Quality attributes . . . . .                            | 29            |
| 1.2.2. Reliability . . . . .                                   | 30            |
| 1.3. Requirements, behavior, and correctness . . . . .         | 31            |
| 1.3.1. Input domain and program correctness . . . . .          | 32            |
| 1.3.2. Valid and invalid inputs . . . . .                      | 33            |
| 1.4. Correctness versus reliability . . . . .                  | 34            |
| 1.4.1. Correctness . . . . .                                   | 34            |
| 1.4.2. Reliability . . . . .                                   | 35            |
| 1.4.3. Program use and the operational profile . . . . .       | 35            |
| 1.5. Testing and debugging . . . . .                           | 36            |
| 1.5.1. Preparing a test plan . . . . .                         | 36            |
| 1.5.2. Constructing test data . . . . .                        | 36            |
| 1.5.3. Executing the program . . . . .                         | 39            |
| 1.5.4. Specifying program behavior . . . . .                   | 40            |
| 1.5.5. Assessing the correctness of program behavior . . . . . | 42            |
| 1.5.6. Construction of oracles . . . . .                       | 44            |
| 1.6. Test metrics . . . . .                                    | 45            |
| 1.6.1. Organizational metrics . . . . .                        | 45            |
| 1.6.2. Project metrics . . . . .                               | 46            |
| 1.6.3. Process metrics . . . . .                               | 46            |

|         |                                                               |    |
|---------|---------------------------------------------------------------|----|
| 1.6.4.  | Product metrics: generic . . . . .                            | 47 |
| 1.6.5.  | Product metrics: OO software . . . . .                        | 47 |
| 1.6.6.  | Progress monitoring and trends . . . . .                      | 49 |
| 1.6.7.  | Static and dynamic metrics . . . . .                          | 49 |
| 1.6.8.  | Testability . . . . .                                         | 49 |
| 1.7.    | Software and hardware testing . . . . .                       | 50 |
| 1.8.    | Testing and verification . . . . .                            | 52 |
| 1.9.    | Defect management . . . . .                                   | 53 |
| 1.10.   | Execution History . . . . .                                   | 54 |
| 1.11.   | Test generation strategies . . . . .                          | 54 |
| 1.12.   | Static testing . . . . .                                      | 56 |
| 1.12.1. | Walkthroughs . . . . .                                        | 57 |
| 1.12.2. | Inspections . . . . .                                         | 57 |
| 1.12.3. | Use of static code analysis tools in static testing . . . . . | 58 |
| 1.12.4. | Software Complexity and static testing . . . . .              | 59 |
| 1.13.   | Model-based testing and model checking . . . . .              | 59 |
| 1.14.   | Control Flow Graph . . . . .                                  | 60 |
| 1.14.1. | Basic block . . . . .                                         | 61 |
| 1.14.2. | Flow graph: definition and pictorial representation . . . . . | 62 |
| 1.14.3. | Path . . . . .                                                | 63 |
| 1.15.   | Dominators and post-dominators . . . . .                      | 67 |
| 1.16.   | Program Dependence Graph . . . . .                            | 69 |
| 1.16.1. | Data dependence . . . . .                                     | 69 |
| 1.16.2. | Control dependence . . . . .                                  | 70 |
| 1.17.   | Strings, languages, and regular expressions . . . . .         | 72 |
| 1.18.   | Types of testing . . . . .                                    | 73 |
| 1.18.1. | Classifier: C1: Source of test generation . . . . .           | 74 |
| 1.18.2. | Classifier: C2: Life cycle phase . . . . .                    | 76 |
| 1.18.3. | Classifier: C3: Goal directed testing . . . . .               | 77 |
| 1.18.4. | Classifier: C4: Artifact under test . . . . .                 | 79 |
| 1.18.5. | Classifier: C5: Test process models . . . . .                 | 80 |
| 1.19.   | The saturation effect . . . . .                               | 84 |
| 1.20.   | Summary . . . . .                                             | 87 |
|         | Bibliographic Notes . . . . .                                 | 88 |
|         | Exercises . . . . .                                           | 93 |

## Part II: Test Generation

97

---

|                                                                       |                |
|-----------------------------------------------------------------------|----------------|
| <b>2. Test Generation: From Requirements</b>                          | <b>98</b>      |
| 2.1. Introduction . . . . .                                           | 98             |
| 2.2. The test selection problem . . . . .                             | 100            |
| 2.3. Equivalence partitioning . . . . .                               | 101            |
| 2.3.1. Faults targeted . . . . .                                      | 102            |
| 2.3.2. Relations and equivalence partitioning . . . . .               | 103            |
| 2.3.3. Equivalence classes for variables . . . . .                    | 107            |
| 2.3.4. Uni-dimensional versus multidimensional partitioning . . . . . | 110            |
| 2.3.5. A systematic procedure for equivalence partitioning . . . . .  | 112            |
| 2.3.6. Test selection based on equivalence classes . . . . .          | 116            |
| 2.3.7. GUI design and equivalence classes . . . . .                   | 119            |
| 2.4. Boundary value analysis . . . . .                                | 120            |
| 2.5. Category-partition method . . . . .                              | 126            |
| 2.6. Cause-effect graphing . . . . .                                  | 132            |
| 2.6.1. Notation used in Cause-effect graphing . . . . .               | 133            |
| 2.6.2. Creating cause-effect graphs . . . . .                         | 135            |
| 2.6.3. Decision table from cause-effect graph . . . . .               | 139            |
| 2.6.4. Heuristics to avoid combinatorial explosion . . . . .          | 143            |
| 2.6.5. Test generation from a decision table . . . . .                | 146            |
| 2.7. Test generation from predicates . . . . .                        | 146            |
| 2.7.1. Predicates and Boolean expressions . . . . .                   | 147            |
| 2.7.2. Fault model for predicate testing . . . . .                    | 148            |
| 2.7.3. Predicate constraints . . . . .                                | 150            |
| 2.7.4. Predicate testing criteria . . . . .                           | 152            |
| 2.7.5. Generating BOR, BRO, and BRE adequate tests . . . . .          | 153            |
| 2.7.6. Cause effect graphs and predicate testing . . . . .            | 166            |
| 2.7.7. Fault propagation . . . . .                                    | 166            |
| 2.7.8. Predicate testing in practice . . . . .                        | 168            |
| 2.8. Summary . . . . .                                                | 171            |
| Bibliographic Notes . . . . .                                         | 171            |
| Exercises . . . . .                                                   | 174            |
| <br><b>3. Test Generation: FSM Models</b>                             | <br><b>181</b> |
| 3.1. Software design and testing . . . . .                            | 181            |
| 3.2. Finite state machines . . . . .                                  | 183            |
| 3.2.1. Excitation using an input sequence . . . . .                   | 186            |
| 3.2.2. Tabular representation . . . . .                               | 187            |
| 3.2.3. Properties of FSM . . . . .                                    | 188            |
| 3.3. Conformance testing . . . . .                                    | 189            |
| 3.3.1. Reset inputs . . . . .                                         | 191            |
| 3.3.2. The testing problem . . . . .                                  | 193            |

---

---

|           |                                                                   |            |
|-----------|-------------------------------------------------------------------|------------|
| 3.4.      | A fault model . . . . .                                           | 193        |
| 3.4.1.    | Mutants of FSMs . . . . .                                         | 195        |
| 3.4.2.    | Fault coverage . . . . .                                          | 197        |
| 3.5.      | Characterization set . . . . .                                    | 198        |
| 3.5.1.    | Construction of the $k$ -equivalence partitions . . . . .         | 198        |
| 3.5.2.    | Deriving the characterization set . . . . .                       | 202        |
| 3.5.3.    | Identification sets . . . . .                                     | 203        |
| 3.6.      | The W-method . . . . .                                            | 204        |
| 3.6.1.    | Assumptions . . . . .                                             | 205        |
| 3.6.2.    | Maximum number of states . . . . .                                | 205        |
| 3.6.3.    | Computation of the transition cover set . . . . .                 | 205        |
| 3.6.4.    | Constructing $Z$ . . . . .                                        | 207        |
| 3.6.5.    | Deriving a test set . . . . .                                     | 207        |
| 3.6.6.    | Testing using the W-method . . . . .                              | 208        |
| 3.6.7.    | The error detection process . . . . .                             | 210        |
| 3.7.      | The partial W-method . . . . .                                    | 211        |
| 3.7.1.    | Testing using the Wp-method for $m = n$ . . . . .                 | 212        |
| 3.7.2.    | Testing using the Wp-method for $m > n$ . . . . .                 | 215        |
| 3.8.      | The UIO-sequence method . . . . .                                 | 217        |
| 3.8.1.    | Assumptions . . . . .                                             | 217        |
| 3.8.2.    | UIO sequences . . . . .                                           | 217        |
| 3.8.3.    | Core and non-core behavior . . . . .                              | 219        |
| 3.8.4.    | Generation of UIO sequences . . . . .                             | 221        |
| 3.8.5.    | Distinguishing signatures . . . . .                               | 231        |
| 3.8.6.    | Test generation . . . . .                                         | 232        |
| 3.8.7.    | Test optimization . . . . .                                       | 234        |
| 3.8.8.    | Fault detection . . . . .                                         | 235        |
| 3.9.      | Automata theoretic versus control-flow based techniques . . . . . | 237        |
| 3.9.1.    | n-switch-cover . . . . .                                          | 240        |
| 3.9.2.    | Comparing automata theoretic methods . . . . .                    | 241        |
| 3.10.     | Summary . . . . .                                                 | 242        |
|           | Bibliographic Notes . . . . .                                     | 243        |
|           | Exercises . . . . .                                               | 246        |
| <b>4.</b> | <b>Test Generation: Statecharts</b>                               | <b>251</b> |
| 4.1.      | Statecharts . . . . .                                             | 251        |
| 4.1.1.    | States . . . . .                                                  | 252        |
| 4.1.2.    | Transitions . . . . .                                             | 253        |
| 4.1.3.    | Pseudo states . . . . .                                           | 254        |
| 4.1.4.    | Events . . . . .                                                  | 258        |
| 4.1.5.    | Actions . . . . .                                                 | 260        |

---

---

|           |                                                                     |            |
|-----------|---------------------------------------------------------------------|------------|
| 4.1.6.    | Statechart and behavior . . . . .                                   | 260        |
| 4.1.7.    | Compound transitions . . . . .                                      | 264        |
| 4.1.8.    | Scope of states and transitions . . . . .                           | 265        |
| 4.2.      | Test generation from Statecharts . . . . .                          | 266        |
| 4.3.      | Statecharts as I/O transformers . . . . .                           | 268        |
| 4.4.      | Distinguishing pairs of states . . . . .                            | 269        |
| 4.5.      | Statecharts with no compound states . . . . .                       | 270        |
| 4.6.      | Test harness and internal events . . . . .                          | 272        |
| 4.7.      | Handling partial labels . . . . .                                   | 274        |
| 4.8.      | Discarding and pruning tests . . . . .                              | 276        |
| 4.9.      | Handling connectors . . . . .                                       | 280        |
| 4.10.     | Handling guards . . . . .                                           | 283        |
| 4.10.1.   | Constructing $P$ and $W$ . . . . .                                  | 284        |
| 4.10.2.   | Generation of variable-value pairs . . . . .                        | 286        |
| 4.11.     | Statecharts with OR states . . . . .                                | 289        |
| 4.12.     | Interlevel transitions . . . . .                                    | 295        |
| 4.13.     | History connectors . . . . .                                        | 297        |
| 4.14.     | Statecharts with AND states . . . . .                               | 298        |
| 4.15.     | Handling forks and merges . . . . .                                 | 302        |
| 4.16.     | Fault detection in statecharts and the need for $W$ -sets . . . . . | 304        |
| 4.17.     | Exploiting hierarchy in test generation . . . . .                   | 306        |
| 4.18.     | Test assessment and enhancement . . . . .                           | 307        |
| 4.18.1.   | Statechart-based assessment . . . . .                               | 308        |
| 4.18.2.   | IUT-based assessment . . . . .                                      | 308        |
| 4.19.     | Summary . . . . .                                                   | 309        |
|           | Bibliographic Notes . . . . .                                       | 309        |
|           | Exercises . . . . .                                                 | 311        |
| <b>5.</b> | <b>Test Generation: Timed I/O Automata</b>                          | <b>321</b> |
| 5.1.      | Introduction . . . . .                                              | 321        |
| 5.2.      | Overview of the test methodology . . . . .                          | 322        |
| 5.3.      | Timed automata . . . . .                                            | 324        |
| 5.3.1.    | Informal introduction . . . . .                                     | 324        |
| 5.3.2.    | Interpretation of clock guards . . . . .                            | 327        |
| 5.3.3.    | Timed trace . . . . .                                               | 329        |
| 5.3.4.    | Clocks and real-time systems . . . . .                              | 329        |
| 5.3.5.    | Time assignments . . . . .                                          | 333        |
| 5.3.6.    | Equivalence of time assignments . . . . .                           | 334        |
| 5.3.7.    | Timed automata and Timed Input/Output Automata . . . . .            | 335        |
| 5.3.8.    | Clock regions . . . . .                                             | 336        |
| 5.3.9.    | Successor regions . . . . .                                         | 338        |

---

---

|                                                                    |            |
|--------------------------------------------------------------------|------------|
| 5.3.10. Region graphs . . . . .                                    | 339        |
| 5.4. Fault model . . . . .                                         | 342        |
| 5.4.1. Transfer and action faults . . . . .                        | 342        |
| 5.4.2. Clock reset faults . . . . .                                | 342        |
| 5.4.3. Faults in the specification of timing constraints . . . . . | 345        |
| 5.5. Tests and delays . . . . .                                    | 347        |
| 5.6. Clock granularity . . . . .                                   | 349        |
| 5.7. Observing the IUT during test . . . . .                       | 351        |
| 5.8. Test generation using TIOA models . . . . .                   | 354        |
| 5.8.1. Grid automata . . . . .                                     | 354        |
| 5.8.2. NTFSMs and generation from grid automata . . . . .          | 358        |
| 5.8.3. Trace equivalence . . . . .                                 | 361        |
| 5.8.4. Delay and input identified states . . . . .                 | 362        |
| 5.8.5. Test Generation using the timed Wp method . . . . .         | 363        |
| 5.8.6. Assumptions that guarantee fault detection . . . . .        | 369        |
| 5.8.7. Mapping faults to NTFSM . . . . .                           | 371        |
| 5.8.8. Fault detection . . . . .                                   | 371        |
| 5.9. Summary . . . . .                                             | 374        |
| Bibliographic Notes . . . . .                                      | 375        |
| Exercises . . . . .                                                | 378        |
| <b>6. Test Generation: Combinatorial Designs</b>                   | <b>383</b> |
| 6.1. Combinatorial designs . . . . .                               | 383        |
| 6.1.1. Test configuration and test set . . . . .                   | 384        |
| 6.1.2. Modeling the input and configuration spaces . . . . .       | 384        |
| 6.2. A combinatorial test design process . . . . .                 | 388        |
| 6.3. Fault model . . . . .                                         | 391        |
| 6.4. Latin Squares . . . . .                                       | 393        |
| 6.5. Mutually orthogonal Latin squares . . . . .                   | 394        |
| 6.6. Pairwise design: binary factors . . . . .                     | 396        |
| 6.7. Pairwise design: multi-valued factors . . . . .               | 401        |
| 6.8. Orthogonal arrays . . . . .                                   | 408        |
| 6.9. Covering and mixed-level covering arrays . . . . .            | 412        |
| 6.10. Arrays of strength $> 2$ . . . . .                           | 413        |
| 6.11. Generating covering arrays . . . . .                         | 414        |
| 6.12. Summary . . . . .                                            | 421        |
| Bibliographic Notes . . . . .                                      | 422        |
| Exercises . . . . .                                                | 424        |

|                                                                                 |            |
|---------------------------------------------------------------------------------|------------|
| <b>7. Test Generation: Regression</b>                                           | <b>428</b> |
| 7.1. What is regression testing ? . . . . .                                     | 428        |
| 7.2. Regression test process . . . . .                                          | 430        |
| 7.2.1. Test revalidation, selection, minimization, and prioritization . . . . . | 430        |
| 7.2.2. Test setup . . . . .                                                     | 431        |
| 7.2.3. Test sequencing . . . . .                                                | 432        |
| 7.2.4. Test execution . . . . .                                                 | 434        |
| 7.2.5. Output comparison . . . . .                                              | 434        |
| 7.3. Regression test selection: the problem . . . . .                           | 434        |
| 7.4. Selecting regression tests . . . . .                                       | 436        |
| 7.4.1. Test all . . . . .                                                       | 436        |
| 7.4.2. Random selection . . . . .                                               | 436        |
| 7.4.3. Selecting modification traversing tests . . . . .                        | 436        |
| 7.4.4. Test minimization . . . . .                                              | 437        |
| 7.4.5. Test prioritization . . . . .                                            | 437        |
| 7.5. Test selection using execution trace . . . . .                             | 438        |
| 7.5.1. Obtaining the execution trace . . . . .                                  | 438        |
| 7.5.2. Selecting regression tests . . . . .                                     | 440        |
| 7.5.3. Handling function calls . . . . .                                        | 444        |
| 7.5.4. Handling changes in declarations . . . . .                               | 445        |
| 7.6. Test selection using dynamic slicing . . . . .                             | 447        |
| 7.6.1. Dynamic slicing . . . . .                                                | 448        |
| 7.6.2. Computation of dynamic slices . . . . .                                  | 449        |
| 7.6.3. Selecting tests . . . . .                                                | 450        |
| 7.6.4. Potential dependence . . . . .                                           | 451        |
| 7.6.5. Computing the relevant slice . . . . .                                   | 454        |
| 7.6.6. Addition and deletion of statements . . . . .                            | 455        |
| 7.6.7. Identifying variables for slicing . . . . .                              | 456        |
| 7.6.8. Reduced dynamic dependence graph . . . . .                               | 456        |
| 7.7. Scalability of test selection algorithms . . . . .                         | 457        |
| 7.8. Test minimization . . . . .                                                | 459        |
| 7.8.1. The set cover problem . . . . .                                          | 460        |
| 7.8.2. A procedure for test minimization . . . . .                              | 461        |
| 7.9. Test prioritization . . . . .                                              | 463        |
| 7.10. Tools for regression testing . . . . .                                    | 466        |
| 7.11. Summary . . . . .                                                         | 467        |
| Bibliographic Notes . . . . .                                                   | 473        |
| Exercises . . . . .                                                             | 474        |

|                                                                  |                |
|------------------------------------------------------------------|----------------|
| <b>8. Test Generation: Formal Specifications</b>                 | <b>479</b>     |
| 8.1. Introduction . . . . .                                      | 479            |
| 8.2. The Z notation in brief . . . . .                           | 481            |
| 8.2.1. A Z specification . . . . .                               | 481            |
| 8.2.2. Sets . . . . .                                            | 481            |
| 8.2.3. Types . . . . .                                           | 483            |
| 8.2.4. Expressions and types . . . . .                           | 483            |
| 8.2.5. Predicates . . . . .                                      | 484            |
| 8.2.6. Sequence . . . . .                                        | 484            |
| 8.2.7. Schema . . . . .                                          | 486            |
| 8.2.8. Combining schemas . . . . .                               | 487            |
| 8.2.9. Operations and features . . . . .                         | 492            |
| 8.3. Test generation using the Test Template framework . . . . . | 493            |
| 8.3.1. Valid input domain . . . . .                              | 494            |
| 8.3.2. Test templates . . . . .                                  | 495            |
| 8.3.3. Template division strategies . . . . .                    | 497            |
| 8.3.4. Domain propagation . . . . .                              | 497            |
| 8.3.5. Number of occurrences . . . . .                           | 499            |
| 8.3.6. Specification mutation . . . . .                          | 500            |
| 8.3.7. Test template instantiation . . . . .                     | 502            |
| 8.3.8. Range of operations and the oracle . . . . .              | 504            |
| 8.4. Robustness testing using formal specifications . . . . .    | 507            |
| 8.5. Digression from formal specifications . . . . .             | 508            |
| 8.6. Summary . . . . .                                           | 509            |
| Bibliographic Notes . . . . .                                    | 509            |
| Exercises . . . . .                                              | 513            |
| <br><b>Part III: Test Adequacy Assessment and Enhancement</b>    | <br><b>517</b> |
| <b>9. Test Adequacy: Control Flow and Data Flow</b>              | <b>518</b>     |
| 9.1. Test adequacy: basics . . . . .                             | 518            |
| 9.1.1. What is test adequacy? . . . . .                          | 518            |
| 9.1.2. Measurement of test adequacy . . . . .                    | 519            |
| 9.1.3. Test enhancement using measurements of adequacy . . . . . | 521            |
| 9.1.4. Infeasibility and test adequacy . . . . .                 | 525            |
| 9.1.5. Error detection and test enhancement . . . . .            | 527            |
| 9.1.6. Single and multiple executions . . . . .                  | 529            |
| 9.2. Adequacy criteria based on control flow . . . . .           | 530            |
| 9.2.1. Statement and block coverage . . . . .                    | 530            |
| 9.2.2. Conditions and decisions . . . . .                        | 532            |
| 9.2.3. Decision coverage . . . . .                               | 534            |



---

|            |                                                          |            |
|------------|----------------------------------------------------------|------------|
| 9.2.4.     | Condition coverage . . . . .                             | 536        |
| 9.2.5.     | Condition/decision coverage . . . . .                    | 538        |
| 9.2.6.     | Multiple condition coverage . . . . .                    | 539        |
| 9.2.7.     | Linear code sequence and jump (LCSAJ) coverage . . . . . | 542        |
| 9.2.8.     | Modified condition/decision coverage . . . . .           | 545        |
| 9.2.9.     | MC/DC adequate tests for compound conditions . . . . .   | 546        |
| 9.2.10.    | Definition of MC/DC coverage . . . . .                   | 550        |
| 9.2.11.    | Minimal MC/DC tests . . . . .                            | 558        |
| 9.2.12.    | Error detection and MC/DC adequacy . . . . .             | 558        |
| 9.2.13.    | Short-circuit evaluation and infeasibility . . . . .     | 560        |
| 9.2.14.    | Tracing test cases to requirements . . . . .             | 561        |
| 9.3.       | Data flow concepts . . . . .                             | 563        |
| 9.3.1.     | Definitions and uses . . . . .                           | 564        |
| 9.3.2.     | C-use and p-use . . . . .                                | 565        |
| 9.3.3.     | Global and local definitions and uses . . . . .          | 566        |
| 9.3.4.     | Data-flow graph . . . . .                                | 566        |
| 9.3.5.     | Def-clear paths . . . . .                                | 568        |
| 9.3.6.     | Def-use pairs . . . . .                                  | 569        |
| 9.3.7.     | Def-use chains . . . . .                                 | 570        |
| 9.3.8.     | A little optimization . . . . .                          | 571        |
| 9.3.9.     | Data contexts and ordered data contexts . . . . .        | 572        |
| 9.4.       | Adequacy criteria based on data flow . . . . .           | 574        |
| 9.4.1.     | <i>c-use</i> coverage . . . . .                          | 575        |
| 9.4.2.     | <i>p-use</i> coverage . . . . .                          | 576        |
| 9.4.3.     | <i>all-uses</i> coverage . . . . .                       | 577        |
| 9.4.4.     | <i>k-dr</i> chain coverage . . . . .                     | 578        |
| 9.4.5.     | Using the <i>k-dr</i> chain coverage . . . . .           | 579        |
| 9.4.6.     | Infeasible c- and p-uses . . . . .                       | 580        |
| 9.4.7.     | Context coverage . . . . .                               | 580        |
| 9.5.       | Control Flow versus Data Flow . . . . .                  | 583        |
| 9.6.       | The “subsumes” Relation . . . . .                        | 585        |
| 9.7.       | Structural and functional testing . . . . .              | 586        |
| 9.8.       | Scalability of coverage measurement . . . . .            | 588        |
| 9.9.       | Summary . . . . .                                        | 590        |
|            | Bibliographic Notes . . . . .                            | 590        |
|            | Exercises . . . . .                                      | 597        |
| <b>10.</b> | <b>Test adequacy: Program Mutation</b>                   | <b>604</b> |
| 10.1.      | Introduction . . . . .                                   | 604        |
| 10.2.      | Mutation and mutants . . . . .                           | 605        |
| 10.2.1.    | First-order and higher-order mutants . . . . .           | 606        |

---

---

|                                                                               |     |
|-------------------------------------------------------------------------------|-----|
| 10.2.2. Syntax and semantics of mutants . . . . .                             | 607 |
| 10.2.3. Strong and weak mutations . . . . .                                   | 610 |
| 10.2.4. Why mutate? . . . . .                                                 | 611 |
| 10.3. Test assessment using mutation . . . . .                                | 612 |
| 10.3.1. A procedure for test adequacy assessment . . . . .                    | 613 |
| 10.3.2. Alternate procedures for test adequacy assessment . . . . .           | 620 |
| 10.3.3. “Distinguished” versus “killed” mutants . . . . .                     | 620 |
| 10.3.4. Conditions for distinguishing a mutant . . . . .                      | 621 |
| 10.4. Mutation operators . . . . .                                            | 622 |
| 10.4.1. Operator types . . . . .                                              | 624 |
| 10.4.2. Language dependence of mutation operators . . . . .                   | 625 |
| 10.5. Design of mutation operators . . . . .                                  | 626 |
| 10.5.1. Goodness criteria for mutation operators . . . . .                    | 626 |
| 10.5.2. Guidelines . . . . .                                                  | 627 |
| 10.6. Founding principles of mutation testing . . . . .                       | 628 |
| 10.6.1. The competent programmer hypothesis . . . . .                         | 628 |
| 10.6.2. The coupling effect . . . . .                                         | 629 |
| 10.7. Equivalent mutants . . . . .                                            | 629 |
| 10.8. Fault detection using mutation . . . . .                                | 630 |
| 10.9. Types of mutants . . . . .                                              | 632 |
| 10.10. Mutation operators for C . . . . .                                     | 633 |
| 10.10.1. What is not mutated? . . . . .                                       | 633 |
| 10.10.2. Linearization . . . . .                                              | 634 |
| 10.10.3. Execution Sequence . . . . .                                         | 636 |
| 10.10.4. Effect of an execution sequence . . . . .                            | 638 |
| 10.10.5. Global and local identifier Sets . . . . .                           | 639 |
| 10.10.6. Global and local reference sets . . . . .                            | 639 |
| 10.10.7. Mutating program constants . . . . .                                 | 642 |
| 10.10.8. Mutating operators . . . . .                                         | 643 |
| 10.10.9. Mutating statements . . . . .                                        | 648 |
| 10.10.10. Mutating program variables . . . . .                                | 660 |
| 10.11. Mutation operators for Java . . . . .                                  | 664 |
| 10.11.1. Traditional mutation operators . . . . .                             | 666 |
| 10.11.2. Inheritance . . . . .                                                | 666 |
| 10.11.3. Polymorphism and dynamic binding . . . . .                           | 670 |
| 10.11.4. Method overloading . . . . .                                         | 671 |
| 10.11.5. Java specific mutation operators . . . . .                           | 672 |
| 10.12. Mutation operators for Fortran 77, C, and Java: a comparison . . . . . | 674 |
| 10.13. Tools for mutation testing . . . . .                                   | 676 |
| 10.14. Mutation testing within budget . . . . .                               | 677 |
| 10.14.1. Prioritizing functions to be mutated . . . . .                       | 677 |

---

|                                                             |     |
|-------------------------------------------------------------|-----|
| 10.14.2. Selecting a subset of mutation operators . . . . . | 678 |
| 10.15. Summary . . . . .                                    | 679 |
| Bibliographic Notes . . . . .                               | 680 |
| Exercises . . . . .                                         | 691 |
| Bibliography . . . . .                                      | 698 |
| Subject Index . . . . .                                     | 746 |
| Name Index . . . . .                                        | 778 |