# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Figures</td>
<td>ix</td>
</tr>
<tr>
<td>List of Tables</td>
<td>xii</td>
</tr>
<tr>
<td>Preface to the Second Edition</td>
<td>xiv</td>
</tr>
<tr>
<td>Part 1 Foundations</td>
<td></td>
</tr>
<tr>
<td>1 Why Do We Test Software?</td>
<td>1</td>
</tr>
<tr>
<td>1.1 When Software Goes Bad</td>
<td>3</td>
</tr>
<tr>
<td>1.2 Goals of Testing Software</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Bibliographic Notes</td>
<td>8</td>
</tr>
<tr>
<td>2 Model-Driven Test Design</td>
<td>19</td>
</tr>
<tr>
<td>2.1 Software Testing Foundations</td>
<td>20</td>
</tr>
<tr>
<td>2.2 Software Testing Activities</td>
<td>21</td>
</tr>
<tr>
<td>2.3 Testing Levels Based on Software Activity</td>
<td>22</td>
</tr>
<tr>
<td>2.4 Coverage Criteria</td>
<td>25</td>
</tr>
<tr>
<td>2.5 Model-Driven Test Design</td>
<td>27</td>
</tr>
<tr>
<td>2.5.1 Test Design</td>
<td>28</td>
</tr>
<tr>
<td>2.5.2 Test Automation</td>
<td>29</td>
</tr>
<tr>
<td>2.5.3 Test Execution</td>
<td>29</td>
</tr>
<tr>
<td>2.5.4 Test Evaluation</td>
<td>29</td>
</tr>
<tr>
<td>2.5.5 Test Personnel and Abstraction</td>
<td>29</td>
</tr>
<tr>
<td>2.6 Why MDTD Matters</td>
<td>31</td>
</tr>
<tr>
<td>2.7 Bibliographic Notes</td>
<td>33</td>
</tr>
<tr>
<td>3 Test Automation</td>
<td>35</td>
</tr>
<tr>
<td>3.1 Software Testability</td>
<td>36</td>
</tr>
<tr>
<td>3.2 Components of a Test Case</td>
<td>36</td>
</tr>
</tbody>
</table>
Contents

3.3 A Test Automation Framework 39
  3.3.1 The JUnit Test Framework 40
  3.3.2 Data-Driven Tests 44
  3.3.3 Adding Parameters to Unit Tests 47
  3.3.4 JUnit from the Command Line 50
3.4 Beyond Test Automation 50
3.5 Bibliographic Notes 53

4 Putting Testing First 54
  4.1 Taming the Cost-of-Change Curve 54
    4.1.1 Is the Curve Really Tamed? 56
  4.2 The Test Harness as Guardian 57
    4.2.1 Continuous Integration 58
    4.2.2 System Tests in Agile Methods 59
    4.2.3 Adding Tests to Legacy Systems 60
    4.2.4 Weaknesses in Agile Methods for Testing 61
  4.3 Bibliographic Notes 62

5 Criteria-Based Test Design 64
  5.1 Coverage Criteria Defined 64
  5.2 Infeasibility and Subsumption 68
  5.3 Advantages of Using Coverage Criteria 68
  5.4 Next Up 70
  5.5 Bibliographic Notes 70

Part 2 Coverage Criteria 73

6 Input Space Partitioning 75
  6.1 Input Domain Modeling 77
    6.1.1 Interface-Based Input Domain Modeling 79
    6.1.2 Functionality-Based Input Domain Modeling 79
    6.1.3 Designing Characteristics 80
    6.1.4 Choosing Blocks and Values 81
    6.1.5 Checking the Input Domain Model 84
  6.2 Combination Strategies Criteria 86
  6.3 Handling Constraints Among Characteristics 92
  6.4 Extended Example: Deriving an IDM from JavaDoc 93
    6.4.1 Tasks in Designing IDM-Based Tests 93
    6.4.2 Designing IDM-Based Tests for Iterator 94
  6.5 Bibliographic Notes 102

7 Graph Coverage 106
  7.1 Overview 106
  7.2 Graph Coverage Criteria 111
    7.2.1 Structural Coverage Criteria 112
    7.2.2 Touring, Sidetrips, and Detours 116
    7.2.3 Data Flow Criteria 123
    7.2.4 Subsumption Relationships Among Graph Coverage Criteria 130
7.3 Graph Coverage for Source Code 131  
   7.3.1 Structural Graph Coverage for Source Code 132  
   7.3.2 Data Flow Graph Coverage for Source Code 136  
7.4 Graph Coverage for Design Elements 146  
   7.4.1 Structural Graph Coverage for Design Elements 147  
   7.4.2 Data Flow Graph Coverage for Design Elements 148  
7.5 Graph Coverage for Specifications 157  
   7.5.1 Testing Sequencing Constraints 157  
   7.5.2 Testing State Behavior of Software 160  
7.6 Graph Coverage for Use Cases 169  
   7.6.1 Use Case Scenarios 171  
7.7 Bibliographic Notes 173  

8 Logic Coverage 177  
8.1 Semantic Logic Coverage Criteria (Active) 178  
   8.1.1 Simple Logic Expression Coverage Criteria 179  
   8.1.2 Active Clause Coverage 181  
   8.1.3 Inactive Clause Coverage 185  
   8.1.4 Infeasibility and Subsumption 186  
   8.1.5 Making a Clause Determine a Predicate 187  
   8.1.6 Finding Satisfying Values 192  
8.2 Syntactic Logic Coverage Criteria (DNF) 197  
   8.2.1 Implicant Coverage 198  
   8.2.2 Minimal DNF 199  
   8.2.3 The MUMCUT Coverage Criterion 200  
   8.2.4 Karnaugh Maps 205  
8.3 Structural Logic Coverage of Programs 208  
   8.3.1 Satisfying Predicate Coverage 212  
   8.3.2 Satisfying Clause Coverage 213  
   8.3.3 Satisfying Active Clause Coverage 215  
   8.3.4 Predicate Transformation Issues 217  
   8.3.5 Side Effects in Predicates 220  
8.4 Specification-Based Logic Coverage 223  
8.5 Logic Coverage of Finite State Machines 226  
8.6 Bibliographic Notes 231  

9 Syntax-Based Testing 234  
9.1 Syntax-Based Coverage Criteria 234  
   9.1.1 Grammar-Based Coverage Criteria 234  
   9.1.2 Mutation Testing 237  
9.2 Program-Based Grammars 241  
   9.2.1 BNF Grammars for Compilers 241  
   9.2.2 Program-Based Mutation 242  
9.3 Integration and Object-Oriented Testing 259  
   9.3.1 BNF Integration Testing 259  
   9.3.2 Integration Mutation 259  
9.4 Specification-Based Grammars 266  
   9.4.1 BNF Grammars 266
Contents

9.4.2 Specification-Based Mutation 267
9.5 Input Space Grammars 271
9.5.1 BNF Grammars 271
9.5.2 Mutating Input Grammars 273
9.6 Bibliographic Notes 281

Part 3 Testing in Practice 283
10 Managing the Test Process 285
10.1 Overview 285
10.2 Requirements Analysis and Specification 286
10.3 System and Software Design 287
10.4 Intermediate Design 288
10.5 Detailed Design 288
10.6 Implementation 289
10.7 Integration 289
10.8 System Deployment 290
10.9 Operation and Maintenance 290
10.10 Implementing the Test Process 291
10.11 Bibliographic Notes 291

11 Writing Test Plans 292
11.1 Level Test Plan Example Template 293
11.2 Bibliographic Notes 295

12 Test Implementation 296
12.1 Integration Order 297
12.2 Test Doubles 298
12.2.1 Stubs and Mocks: Variations of Test Doubles 299
12.2.2 Using Test Doubles to Replace Components 300
12.3 Bibliographic Notes 303

13 Regression Testing for Evolving Software 304
13.1 Bibliographic Notes 306

14 Writing Effective Test Oracles 308
14.1 What Should Be Checked? 308
14.2 Determining Correct Values 310
14.2.1 Specification-Based Direct Verification of Outputs 310
14.2.2 Redundant Computations 311
14.2.3 Consistency Checks 312
14.2.4 Metamorphic Testing 312
14.3 Bibliographic Notes 314

List of Criteria 316
Bibliography 318
Index 337