VERIFICATION AND VALIDATION FOR QUALITY OF UML 2.0 MODELS
VERIFICATION AND VALIDATION FOR QUALITY OF UML 2.0 MODELS

BHUVAN UNHELKAR, PhD
SONKI
Contents

Figures xix
Foreword xxiii
Preface xxv
Acknowledgments xxxi
Glossary of Acronyms and Terms xxxiii
Author Profile xxxv

1 The Quality Strategy for UML 1

Chapter Summary 1

1.1 Modeling and Quality 2
1.1.1 The Modeling Advantage 2
1.1.2 Modeling Caveats 2
1.1.3 Context of Model Quality 3
1.1.4 Model Quality 4

1.2 Positioning UML for Modeling 4

1.3 Quality Aspects of UML 5

1.4 Understanding Modeling Spaces in Software 7

1.5 Modeling Spaces and UML 7
1.5.1 Importance of UML Diagrams to Respective Models 8
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5.2</td>
<td>List of UML Diagrams</td>
<td>8</td>
</tr>
<tr>
<td>1.5.3</td>
<td>UML Diagrams and Modeling Spaces</td>
<td>9</td>
</tr>
<tr>
<td>1.5.4</td>
<td>Model of Problem Space (MOPS)</td>
<td>10</td>
</tr>
<tr>
<td>1.5.5</td>
<td>Model of Solution Space (MOSS)</td>
<td>11</td>
</tr>
<tr>
<td>1.5.6</td>
<td>Model of Background Space (MOBS)</td>
<td>12</td>
</tr>
<tr>
<td>1.6</td>
<td>Verification and Validation</td>
<td>14</td>
</tr>
<tr>
<td>1.6.1</td>
<td>Quality Models—Syntax</td>
<td>15</td>
</tr>
<tr>
<td>1.6.2</td>
<td>Quality Models—Semantics</td>
<td>17</td>
</tr>
<tr>
<td>1.6.3</td>
<td>Quality Models—Aesthetics</td>
<td>18</td>
</tr>
<tr>
<td>1.6.4</td>
<td>Quality Techniques and V&amp;V Checks</td>
<td>18</td>
</tr>
<tr>
<td>1.7</td>
<td>Quality Checks and Skills Levels</td>
<td>19</td>
</tr>
<tr>
<td>1.8</td>
<td>Levels of Quality Checks to UML Diagrams</td>
<td>20</td>
</tr>
<tr>
<td>1.8.1</td>
<td>Syntax Checks and UML Elements (Focus on Correctness)</td>
<td>21</td>
</tr>
<tr>
<td>1.8.2</td>
<td>Semantic Checks and UML Diagrams (Focus on Completeness and Consistency)</td>
<td>22</td>
</tr>
<tr>
<td>1.8.3</td>
<td>Aesthetic Checks and UML Models (Focus on Symmetry and Consistency)</td>
<td>22</td>
</tr>
<tr>
<td>1.9</td>
<td>Model-Driven Architecture (MDA) and Quality</td>
<td>23</td>
</tr>
<tr>
<td>1.10</td>
<td>Prototyping and Modeling Spaces</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td>Discussion Topics</td>
<td>24</td>
</tr>
<tr>
<td></td>
<td>References</td>
<td>25</td>
</tr>
</tbody>
</table>

2 Nature and Basics of UML Diagrams | 27

Chapter Summary | 27

2.1 The Nature of UML Diagrams | 27
| 2.1.1 | Elasticity of UML | 28 |
| 2.1.2 | Structural versus Behavioral Nature of UML Diagrams | 28 |
| 2.1.3 | Static versus Dynamic Nature of UML Diagrams | 29 |

2.2 Use Case Diagrams | 30
| 2.2.1 | Nature of Use Case Diagrams | 30 |
| 2.2.2 | Putting Together a Use Case Diagram | 31 |

2.3 Activity Diagrams | 33
| 2.3.1 | Nature of Activity Diagrams | 33 |
| 2.3.2 | Putting Together an Activity Diagram | 33 |
| 2.3.3 | Specifications in an Activity Diagram | 34 |
3 Strengths, Weaknesses, Objectives and Traps (SWOT) of UML Diagrams

Chapter Summary

3.1 SWOT Analysis of the UML Diagrams

3.2 SWOT of Use Case Diagrams
  3.2.1 Strengths of Use Cases and Use Case Diagrams
  3.2.2 Weaknesses of Use Cases and Use Case Diagrams
  3.2.3 Objectives of Use Cases and Use Case Diagrams
  3.2.4 Traps of Use Cases and Use Case Diagrams

3.3 SWOT of Activity Diagrams
  3.3.1 Strengths of Activity Diagrams
  3.3.2 Weaknesses of Activity Diagrams
  3.3.3 Objectives of Activity Diagrams
  3.3.4 Traps of Activity Diagrams

3.4 SWOT of Classes and Class Diagrams
  3.4.1 Strengths of Classes and Class Diagrams
  3.4.2 Weaknesses of Classes and Class Diagrams
  3.4.3 Objectives of Classes and Class Diagrams
  3.4.4 Traps of Classes and Class Diagrams

3.5 SWOT of Sequence Diagrams
  3.5.1 Strengths of Sequence Diagrams
  3.5.2 Weaknesses of Sequence Diagrams
  3.5.3 Objectives of Sequence Diagrams
  3.5.4 Traps of Sequence Diagrams

3.6 SWOT of Communication Diagrams
  3.6.1 Strengths of Communication Diagrams
  3.6.2 Weaknesses of Communication Diagrams
  3.6.3 Objectives of Communication Diagrams
  3.6.4 Traps of Communication Diagrams
3.7 SWOT of Interaction Overview Diagrams
   3.7.1 Strengths of Interaction Overview Diagrams
   3.7.2 Weaknesses of Interaction Overview Diagrams
   3.7.3 Objectives of Interaction Overview Diagrams
   3.7.4 Traps of Interaction Overview Diagrams

3.8 SWOT of Object Diagrams
   3.8.1 Strengths of Object Diagrams
   3.8.2 Weaknesses of Object Diagrams
   3.8.3 Objectives of Object Diagrams
   3.8.4 Traps of Object Diagrams

3.9 SWOT of State Machine Diagrams
   3.9.1 Strengths of State Machine Diagrams
   3.9.2 Weaknesses of State Machine Diagrams
   3.9.3 Objectives of State Machine Diagrams
   3.9.4 Traps of State Machine Diagrams

3.10 SWOT of Composite Structure Diagrams
   3.10.1 Strengths of Composite Structure Diagrams
   3.10.2 Weaknesses of Composite Structure Diagrams
   3.10.3 Objectives of Composite Structure Diagrams
   3.10.4 Traps of Composite Structure Diagrams

3.11 SWOT of Component Diagrams
   3.11.1 Strengths of Component Diagrams
   3.11.2 Weaknesses of Component Diagrams
   3.11.3 Objectives of Component Diagrams
   3.11.4 Traps of Component Diagrams

3.12 SWOT of Deployment Diagrams
   3.12.1 Strengths of Deployment Diagrams
   3.12.2 Weaknesses of Deployment Diagrams
   3.12.3 Objectives of Deployment Diagrams
   3.12.4 Traps of Deployment Diagrams

3.13 SWOT of Package Diagrams
   3.13.1 Strengths of Package Diagrams
   3.13.2 Weaknesses of Package Diagrams
   3.13.3 Objectives of Package Diagrams
   3.13.4 Traps of Package Diagrams

3.14 SWOT of Timing Diagrams
   3.14.1 Strengths of Timing Diagrams
   3.14.2 Weaknesses of Timing Diagrams
   3.14.3 Objectives of Timing Diagrams
   3.14.4 Traps of Timing Diagrams
4 V&V of the Quality of MOPS

Chapter Summary

4.1 UML Diagrams in MOPS

4.2 V&V of Use Cases and Use Case Diagrams in MOPS

4.2.1 Actors and Use Cases—Initial List

4.2.2 List of Use Case Diagrams

4.2.3 Describing Client Maintenance Use Case Diagrams

4.2.4 Actor–Class Confusion

4.2.5 Actor Documentation and Quality Checks

4.2.6 Describing Policy Creation and Policy Maintenance Use Case Diagrams

4.2.7 Use Case Documentation and Quality Checks

4.2.8 Describing the Claims Processing Use Case Diagram

4.2.9 Describing the Sales Campaigning Use Case Diagram

4.2.10 Syntax Checks for Use Case Diagrams

4.2.11 Semantic Checks for Use Case Diagrams

4.2.12 Aesthetic Checks for Use Case Diagrams

4.2.13 Acceptance Testing and Use Case Documentation

4.3 Quality of Activity Diagrams in MOPS

4.3.1 Describing the AddsClientDetails Activity Diagram

4.3.2 Describing the CreatesHomeInsurancePolicy Activity Diagram

4.3.3 Syntax Checks for Activity Diagrams

4.3.4 Semantic Checks for Activity Diagrams

4.3.5 Aesthetic Checks for Activity Diagrams

4.4 Quality of Package Diagrams in MOPS

4.4.1 Describing the LUCKY Package Diagram

4.4.2 Syntax Checks for the LUCKY Package Diagram

4.4.3 Semantic Checks for the LUCKY Package Diagram

4.4.4 Aesthetic Checks for the LUCKY Package Diagram

4.5 Quality of Classes and Class Diagrams in MOPS

4.5.1 Documenting a Class

4.5.2 Syntax Checks for Classes

4.5.3 Semantic Checks for Classes

4.5.4 Aesthetic Checks for the Client Class
4.5.5 Describing the ClientDetails Class Diagram 139
4.5.6 Describing the PolicyDetails Class Diagram 142
4.5.7 Syntax Checks for Class Diagrams 142
4.5.8 Semantic Checks for Class Diagrams 144
4.5.9 Aesthetic Checks for Class Diagrams 144

4.6 Quality of Sequence Diagrams in MOPS 145
4.6.1 Describing the CreateClient Sequence Diagram 145
4.6.2 Describing the CreateClientOnInternet Sequence Diagram 146
4.6.3 Describing the ApprovePolicy Sequence Diagram 147
4.6.4 Syntax Checks for Sequence Diagrams 148
4.6.5 Semantic Checks for Sequence Diagrams 148
4.6.6 Aesthetic Checks for Sequence Diagrams 150

4.7 Quality of State Machine Diagrams in MOPS 150
4.7.1 Describing the Client State Machine Diagram 150
4.7.2 Describing the Policy State Machine Diagram 151
4.7.3 Syntax Checks for the ClientStates State Machine Diagram 152
4.7.4 Semantic Checks for the ClientStates State Machine Diagram 153
4.7.5 Aesthetic Checks for the ClientStates State Machine Diagram 153

4.8 Quality of Interaction Overview Diagrams in MOPS 154
4.8.1 Describing the CreateClient Interaction Overview Diagram 154
4.8.2 Syntax Checks for Interaction Overview Diagrams 155
4.8.3 Semantic Check for Interaction Overview Diagrams 155
4.8.4 Aesthetic Check for Interaction Overview Diagrams 155

4.9 Validating the Entire MOPS 156
4.9.1 Use Case to Activity Diagram Dependency and Quality 156
4.9.2 Use Case to Class Diagram Dependency and Quality 157
4.9.3 Class to Sequence Diagram Dependency and Quality 157
4.9.4 Interaction Overview Diagram to Sequence Diagram and Use Case Dependencies 157
4.9.5 Quality of Documentation Associated with MOPS 158
4.9.6 Aesthetics of MOPS 158

4.10 Summary of Quality Checks for MOPS 158

Discussion Topics 158

References 160
5 V&V of the Quality of MOSS

Chapter Summary

5.1 UML Diagrams in the Solution Space (MOSS)

5.2 Analyzing MOPS for MOSS for a Solution
   5.2.1 Analysis of Use Cases in the Solution Space
   5.2.2 Analysis of Business Class Diagrams in the Solution Space
   5.2.3 Analyzing Activity Diagrams in the Solution Space

5.3 Quality of Classes and Class Diagrams in MOSS
   5.3.1 Syntax Checks for Classes in the Solution Space
   5.3.2 Semantic Checks for Classes in the Solution Space
   5.3.3 Aesthetic Checks for Classes at the Design Level
   5.3.4 Describing Class Diagrams in MOSS
   5.3.5 Syntax Checks of Class Diagrams in MOSS
   5.3.6 Semantic Checks of Class Diagrams in MOSS
   5.3.7 Aesthetic Checks of Class Diagrams in MOSS

5.4 Quality of Sequence Diagrams in MOSS
   5.4.1 Describing the Sequence of Submission of a Claim
   5.4.2 Syntax Checks for Sequence Diagrams
   5.4.3 Semantic Checks for Sequence Diagrams
   5.4.4 Aesthetic Checks for Sequence Diagrams

5.5 Quality of Communication Diagrams in MOSS
   5.5.1 Describing the SubmitsClaim Communication Diagram
   5.5.2 Syntax Checks for Communication Diagrams
   5.5.3 Semantic Checks for Communication Diagrams
   5.5.4 Aesthetic Checks for Communication Diagrams

5.6 Quality of Object Diagrams in MOSS
   5.6.1 Describing the Object Diagram for Policy and Claim
   5.6.2 Syntax Checks for Object Diagrams
   5.6.3 Semantic Checks for Object Diagrams
   5.6.4 Aesthetic Checks for Object Diagrams

5.7 Quality of State Machine Diagrams in MOSS
   5.7.1 Describing an Advanced State Machine Diagram for Claim
   5.7.2 Syntax Checks for State Machine Diagrams in MOSS
5.7.3 Semantic Checks for State Machine Diagrams in MOSS 187
5.7.4 Aesthetic Checks for State Machine Diagrams in MOSS 188

5.8 Quality of Timing Diagrams in MOSS 188
5.8.1 Describing the Timing Diagram for Policy and Claim in MOSS 188
5.8.2 Syntax Checks for Timing Diagrams in MOSS 188
5.8.3 Semantic Checks for Timing Diagrams in MOSS 188
5.8.4 Aesthetic Checks for Timing Diagrams in MOSS 189

5.9 Converting Models into Systems 189
5.10 Cross-Diagram Dependencies 189

Discussion Topics 190
References 192

6 V&V of the Quality of MOBS 193

Chapter Summary 193
6.1 Working in the Background Space 194
6.2 UML Diagrams in the Background Space (MOBS) 194
   6.2.1 Layers in Software Architecture (Functional vs. Infrastructure) 196
   6.2.2 Relating the Functional Slices to the Infrastructure 198
6.3 V&V of Package Diagrams in MOBS 199
   6.3.1 Syntax Checks for Package Diagrams in MOBS 202
   6.3.2 Semantic Checks for Package Diagrams in MOBS 202
   6.3.3 Aesthetic Checks for Package Diagrams in MOSS 203
6.4 Classes and Class Diagrams in the Background Space 204
   6.4.1 Relational Table Representation by Classes 204
   6.4.2 Mapping ClientDetails to Relational Tables 205
   6.4.3 Describing Active Classes in MOBS 205
   6.4.4 Class and Granularity of Design in the Background Space 207
   6.4.5 Assigning Classes to Components 208
6.5 V&V of Class Diagrams in the Background Space 208
   6.5.1 Syntax Checks for Class Diagrams in MOBS 209
   6.5.2 Semantic Checks for Class Diagrams in MOBS 209
   6.5.3 Aesthetic Checks for Class Diagrams in MOBS 210
Figures

Figure 1.1  Software modeling spaces and the modeling work of analysis, design, and architecture in them.
Figure 1.2  Quality and V&V of models and their mapping to syntax, semantics and aesthetics.
Figure 1.3  Application of syntax, semantics and aesthetics.
Figure 1.4  V&V checks and quality techniques.
Figure 1.5  Syntax, semantics and aesthetics checks verify and validate UML artifacts, diagrams and models.

Figure 2.1  Summary of the nature of UML diagrams, depicting their structural versus behavioral characteristics as well as their static versus dynamic nature.
Figure 2.2  Nature and basics of a use case diagram.
Figure 2.3  Nature and basics of an activity diagram.
Figure 2.4  Nature and basics of a class diagram.
Figure 2.5  Nature and basics of a sequence diagram.
Figure 2.6  Nature and basics of a communication diagram.
Figure 2.7  Nature and basics of an interaction overview diagram.
Figure 2.8  Nature and basics of an object diagram.
Figure 2.9  Nature and basics of a state machine diagram.
Figure 2.10  Nature and basics of a composite structure diagram.
Figure 2.11  Nature and basics of a component diagram.
Figure 2.12  Nature and basics of a deployment diagram.
Figure 2.13  Nature and basics of a package diagram.
Figure 2.14  Nature and basics of a timing diagram.
Figure 2.15  Stereotypes.
Figure 2.16  Notes (A, B and C).
Figure 2.17  Constraints (A, B).
Figure 2.18  Tagged values.
Figure 2.19 Practical application of OMG’s meta-models to improve the quality of UML diagrams.
Figure 2.20 Example of extending the UML meta-model to enforce quality controls.
Figure 3.1 SWOT analysis of UML diagrams.
Figure 4.1 Primary UML diagrams in MOPS.
Figure 4.2 Insurance requirements—list of actors.
Figure 4.3 Client maintenance (a) use case diagram.
Figure 4.4 Client maintenance (b) use case diagram.
Figure 4.5 Client maintenance (c) use case diagram.
Figure 4.6 Actor–class confusion.
Figure 4.7 Policy creation use case diagram.
Figure 4.8 Policy maintenance use case diagram.
Figure 4.9 Claims processing use case diagram.
Figure 4.10 Sales campaigning use case diagram.
Figure 4.11 AddsClientDetails activity diagram.
Figure 4.12 CreatesHomeInsurancePolicy activity diagram.
Figure 4.13 The LUCKY package diagram.
Figure 4.14 Classes Client and Policy.
Figure 4.15 ClientDetails—a class diagram.
Figure 4.16 ClientDetails–b class diagram.
Figure 4.17 ClientDetails–c class diagram.
Figure 4.18 PolicyDetails class diagram.
Figure 4.19 Creation of client sequence diagram.
Figure 4.20 CreateClientOnInternet sequence diagram.
Figure 4.21 Approvepolicy sequence diagram.
Figure 4.22 Client state machine diagram.
Figure 4.23 Policy state machine diagram.
Figure 4.24 Interaction overview diagram for CreateClient.
Figure 5.1 Primary UML diagrams in MOSS.
Figure 5.2 Analysis of the claims processing use case diagram in the solution space.
Figure 5.3 Some classes revealed by the SubmitsClaim use case.
Figure 5.4 Analyzing the SubmitsClaim activity diagram (for the corresponding use case).
Figure 5.5 Upgrading class operations (behavior) by the previous analysis.
Figure 5.6 Addition of design-level classes in MOSS (GUI, database, utility, etc.).
Figure 5.7 SubmitsClaim basic sequence diagram.
Figure 5.8 SubmitsClaim advanced sequence diagram.
Figure 5.9 Creating a SubmitsClaim communication diagram.
Figure 5.10 An object diagram for Policy and Claim.
Figure 5.11 Advanced state machine diagram for Claim in the solution space.
Figure 5.12 Timing diagram for Policy and Claim in the solution space.
Figure 6.1 UML diagrams in the background space.
Figure 6.2 Application- versus infrastructure-based layering in software architecture for the LUCKY insurance system.
Figure 6.3 Putting application architecture in the context of the infrastructure (background) architecture.
Figure 6.4  Insurance system package diagram.
Figure 6.5  Package diagrams as layered frameworks.
Figure 6.6  Mapping classes related to Client to the corresponding relational tables.
Figure 6.7  Considering active classes, processes and threads in MOBS.
Figure 6.8  Granularity of OO designs.
Figure 6.9  Assigning classes to components for physical implementation.
Figure 6.10  Robustness diagrams for PolicyCreation and ClaimsProcessing processes with <<control>>, <<boundary>> and <<entity>> classes.
Figure 6.11  Effect of robustness on the SubmitsClaim sequence diagram.
Figure 6.12  LUCKY insurance components, packages and interfaces.
Figure 6.13  Insurance system component diagrams.
Figure 6.14  An example composite structure diagram for the LUCKY insurance system.
Figure 6.15  Deployment diagram for the LUCKY insurance system.
Figure 6.16  Web Infrastructure in development and deployment of systems.

Figure 7.1  Process wraps around UML.
Figure 7.2  Architecture of a process component.
Figure 7.3  A process component for V&V of classes and class diagrams in MOPS.
Figure 7.4  Iterations and increments (a high-level example).
Foreword

UML (Unified Modeling Language) has now been in use for several years and is currently coming close to the end of a large-scale review and reformulation resulting this year (2004) in a new major version (2.0). With its previous versions (1.1, 1.3, 1.4 and 1.5), UML has been widely used by business analysts, system designers, system architects and testers. However, despite this increasing usage, many companies are not taking the best advantage of UML, and occasionally, individuals have experienced frustration in applying its standards.

Perhaps this is because they have not yet read this book!

In this book, Dr. Unhelkar takes a practical viewpoint in explaining UML, underpinned by a good understanding of the theoretical basis of what makes a quality UML model. He introduces a wide range of guidelines to increase and test the quality of UML models in their various guises. UML users now have the chance to benchmark their own models against an experienced modeler’s hard-won experience and expertise. Unhelkar has been using and teaching UML to industry (and academic) groups since UML began and has previously published books describing UML. In this book, he views UML from a quality perspective, evaluating the strengths and weaknesses of the diagram suite and then offering tips and guidelines on how to approach the quality verification and validation (V&V) checklists, followed later in the book.

A major element of this book is the SWOT analysis (strengths, weaknesses, objectives and traps). This analysis is done on each UML diagram in turn so that the reader can see both the up- and downsides of each diagram. Very importantly, the traps should be studied since Unhelkar gathers together many of the pitfalls well known to those of us who are using and teaching UML but that never before have been written down and made public. The subsequent V&V is undertaken with three foci: syntax, semantics and aesthetics. The first two are fairly
straightforward and can be gleaned from a careful reading of the OMG standard on UML. But nowhere in the standard will you find advice on stylistic use of UML. As the author notes, an aesthetically pleasing diagram can aid understanding. For instance, sensible naming of classes is long established as an aid to communication; conversely, sloppy naming is virtually guaranteed to make a design unusable. While this quality analysis is not quantitative (perhaps Unhelkar’s next book will be the metricated version?), this is an important first step toward that achievement. Nowhere else will you find the quality-focussed insights as presented here into how the UML really functions.

Another feature of the presentation is the delineation of three viewpoints, termed the model of problem space (MOPS), model of solution space (MOSS) and model of background space (MOBS). These models allow the reader to create frameworks in which, depending on the current life cycle stage (e.g., MOSS rather than MOPS), a different mix and balance of UML diagrams is advocated. Again, most UML books do not emphasize this well but rather offer UML as a one-size-fits-all approach, which in many ways it originally was. This, of course, all hints at process, which keeps creeping in, although this is dealt with in detail elsewhere in another of the author’s books.

All of this analysis is held together by a running example of the LUCKY insurance system. This is an excellent vehicle for demonstrating the basic features of UML in early chapters and then for presenting the qualitative quality analysis in later chapters. This means that the reader can see how all the pieces fit together—which is sometimes hard to observe in examples presented in other venues.

The practical checklists are helpful in verifying and validating the UML models well enough to deserve prominence in a software developer’s mindset. I hope that this book will facilitate and encourage such a mindset and recommend that the path to this mindset begins with its perusal.

Brian Henderson-Sellers
Director, Centre for Object Technology
Applications and Research
University of Technology, Sydney
Preface

Only those who detest working on the same thing twice, produce quality!*

PURPOSE

This book presents reasons and techniques to enhance the quality of software models built using the Unified Modeling Language (UML Version 2.0). While models in general enhance software quality, the quality of the models themselves is rarely scrutinized. This book addresses this lacuna by delving deeply into the verification and validation (V&V) of UML 2.0 models. A simple and practical checklist-based approach gives the UML models the attention and rigor they deserve.

UML is primarily a means to communicate between various parties involved in a software project. It can be used by users and business analysts to specify requirements, by designers to produce solution designs and by architects to produce system architecture for deployment of the software solution. As with most communication mechanisms, the effectiveness of UML depends on its accuracy and standardization together with the prevailing conventions and mutual understanding of its usage within the project. These are some of the aspects of the UML discussed in this book. The discussions start by developing an understanding the nature of UML models and how to create good-quality models, followed by the creation and application of significant checks and cross-checks to these models. These checks, discussed in terms of V&V checks in this work, encompass the syntactic

*Source: a lazy brain!
While UML comprises a suite of diagrams, practically it is helpful to consider these diagrams akin to a suite of tools in a toolbox. Viewing UML as a toolbox permits modelers to choose the right diagrams for the role they must play in the modeling spaces. Subsequently, all of the UML diagrams are discussed in this book according to the modeling spaces in which they are most relevant and effective: use cases, activity and business class diagrams, for example, make perfect sense in creating the model of the problem space (MOPS); advanced class diagrams, advanced sequence diagrams and composite structure diagrams are more relevant in creating the model of the solution space (MOSS) than in the other two modeling spaces; and the model of the background space (MOBS), which deals with the architectural and operational requirements of the system, produces maximum advantage by using the component and deployment diagrams. This distribution of the UML diagrams in the modeling spaces also leads to an appropriate distribution of the V&V checks in the three modeling spaces, making it easy and practical for readers to focus on the UML diagrams and their quality checks corresponding to their respective modeling spaces.

CONTENTS

This book is divided into seven chapters and six appendices.

Chapter 1 discusses the concept of quality within the context of UML. This chapter also considers quality levels, modeling spaces, syntax, semantics and aesthetic checks, as well as the corresponding skill levels required for the V&V activities. Although this is more than a basic book on UML, it provides sufficient discussion on the nature and creation of the UML diagrams to enable readers to use it as a self-sufficient resource. That discussion, introduced in Chapters 2 and 3, is designed to create good-quality UML diagrams in the first place. Chapter 4 deals with the V&V checks in the problem space as relevant to a business analyst; Chapter 5 deals with quality checks in the solution space, as relevant to a system designer; and Chapter 6 deals with the quality checks in the architectural (background) modeling space, as relevant to system architect. Chapter 7 discusses the process requirements in the context of the V&V techniques discussed in the earlier chapters. Each chapter is supported by Discussion Topics at the end, not only to enable creative discussions, but also to open up potential research directions.

The core V&V chapters of this book—Chapters 4, 5 and 6—are built around a case study based on the author’s practical experience in modeling. The problem statement for the case study appears separately Appendix A. Appendix C, D and E summarize the checklists from each of the aforementioned chapters, respectively. Readers keen to practice material from this book may even want to jump ahead and start using these checklists directly in their work. Since UML-based CASE tools play an important part in creating good-quality UML diagrams, their role and selection criteria are mentioned in Appendix B.
### SUMMARY OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. The Quality Strategy for UML</td>
<td>This chapter discusses the strategic approach to quality. Divides the UML diagrams in terms of relevance and importance in the three modeling spaces: problem, solution and background. Describes the types and levels of quality checks: syntax, semantics and aesthetics.</td>
</tr>
<tr>
<td>2. Nature and Basics of UML diagrams</td>
<td>This chapter describes all the UML 2.0 diagrams and their nature (structural versus behavioral versus dynamic). The basics of putting these diagrams together is also discussed from a quality perspective.</td>
</tr>
<tr>
<td>3. Strengths, Weaknesses, Objectives and Traps (SWOT) of UML Diagrams</td>
<td>This chapter incorporates a unique feature of this book considered invaluable by many reviewers in understanding the UML diagrams. Discussed here are the strengths and weaknesses (inherent characteristics) of each of the UML diagrams together with their practical (extrinsic characteristics) objectives and traps.</td>
</tr>
<tr>
<td>4. V&amp;V of the Quality of MOPS</td>
<td>This chapter undertakes detailed V&amp;V procedures to ensure and enhance the quality of UML diagrams that comprise MOPS.</td>
</tr>
<tr>
<td>5. V&amp;V of the Quality of MOSS</td>
<td>This chapter undertakes detailed V&amp;V procedures to ensure and enhance the quality of UML diagrams that comprise MOSS.</td>
</tr>
<tr>
<td>6. V&amp;V of the Quality of MOBS</td>
<td>This chapter undertakes detailed V&amp;V procedures to ensure and enhance the quality of UML diagrams that comprise MOBS.</td>
</tr>
<tr>
<td>7. Managing the V&amp;V Process</td>
<td>This chapter discusses the process aspect of the V&amp;V undertaken in the previous three chapters. Discussion includes roles and responsibilities, deliverables and activities, and tasks performed during quality assurance.</td>
</tr>
</tbody>
</table>

This book is also supported by six appendices, as summarized in the following table.
Appendix Description
Appendix A LUCKY Insurance System case study problem statement. Forms basis of models in Chapters 4, 5 and 6.
Appendix B Assessment of UML Case Tools
Appendix C Practical checklists summary for MOPS
Appendix D Practical checklists summary for MOSS
Appendix E Practical checklists summary for MOBS
Appendix F Templates for actors, use cases and classes in MOPS

Web Support
Relevant parts of this book are made available through the Web site of John Wiley and Sons (www.wiley.com).

AUDIENCE
This book aims to be an indispensable resource to quality professionals including quality analysts, process consultants, quality managers, test designers and testers. Equally important is the value this material provides to the three central roles in the modeling spaces: business analysts (quality in the problem modeling space), system designers (quality in the solution modeling space) and system architects (quality in the background modeling space). In addition, the philosophical discussions on modeling and quality in Chapter 1, as well as the process discussion in Chapter 7, will be of interest to senior project managers and directors. As a result, the book succeeds in providing practical support in V&V of UML models.

In addition to its practical applicability, this book has a rich academic and research base emanating from a number of enthusiastic and brilliant research associates surrounding the author. The extensive referencing in the relevant chapters, as well as the discussion topics, will be invaluable to students and researchers in quality, modeling and UML.

This is an intermediate to advanced book dealing with the topic of quality and UML. Hence the reader is expected to be familiar with UML and its modeling techniques, as the book does not discuss the basics of UML itself. People responsible for quality assurance will find this work self-sufficient and may even be encouraged, after reading it, to further extend their understanding to UML.

LANGUAGE
I firmly believe in gender-neuter language. “Person” is therefore used wherever possible. However, in order to maintain simplicity of reading, “he” has been used as freely as “she.”