Undergraduate Topics in Computer Science

Martina Seidl • Marion Scholz
Christian Huemer • Gerti Kappel

UML @ Classroom

An Introduction to Object-Oriented Modeling

Springer

Martina Seidl
Johannes Kepler University Linz
Linz, Austria

Marion Scholz
Vienna University of Technology
Vienna, Austria

Christian Huemer
Vienna University of Technology
Vienna, Austria

Gerti Kappel
Vienna University of Technology
Vienna, Austria

Tanslator
Tracey Duffy
TSD Translations

Copyright © 2012 by dpunkt.verlag GmbH, Heidelberg, Germany.
Title of the German original: UML @ Classroom
ISBN 978-3-89864-776-2
Translation Copyright © 2014 by Springer International Publishing AG.
All rights reserved.

ISSN 1863-7310 ISSN 2197-1781 (electronic)
Undergraduate Topics in Computer Science
ISBN 978-3-319-12741-5 ISBN 978-3-319-12742-2 (eBook)
DOI 10.1007/978-3-319-12742-2

Library of Congress Control Number: 2015930192

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Preface

The challenges in today's software development are diverse and go far beyond implementation tasks. They range from requirement specification over system design and implementation to maintenance and further adaptation of the software—to name just a few phases in the software life cycle. In all of these phases of the software development process, many people with different backgrounds and experiences are usually involved. These people need a common language for efficient communication. Obviously, such a language should be as precise as possible without the ambiguities of a natural language. For this purpose, modeling languages have emerged. They are used to create sketches and blueprints for software systems, which in turn serve as a basis for the implementation or even automatic generation of executable code. In the area of object-oriented software development, the Unified Modeling Language (UML) was able to prevail. Of course, to use the language correctly and efficiently, it is necessary to understand the concepts offered by UML. Since 2006, we have offered the course "Object-Oriented Modeling" at the Vienna University of Technology. This course is mandatory for computer science and business informatics students in their first year. Overall, we have up to 1,000 students per year who attend our course. To deal with such a huge number of students while keeping high quality standards, much effort has been spent on the preparation of such a course. This includes the overall organization, course material, and e-learning support. Parts of the course design have been presented at the Educators' Symposium of the MODELS conference [8, 9, 10, 11, 7, 46]. We teach the basics of object-oriented modeling by means of UML.

In particular, we teach

- class and object diagrams,
- sequence diagrams,
- state machine diagrams,
- activity diagrams, and
- use case diagrams

as well as their interrelations. For this purpose, we introduce the syntax (the notation of the language elements), the semantics (the meaning of the language elements), and the pragmatics (how to use the language elements) of these UML diagrams. They cover the most essential concepts of object-oriented modeling and are used in many different stages of the software development process. The course is designed for students who already know the basic concepts of object-oriented programming languages such as Java or C#, but still have no practical experience in software engineering. Based on our comprehensive experience in teaching UML, we wrote the book UML@Classroom. In this book, we address both readers who wish to learn UML in a compact but nevertheless precise manner and teachers whom we want to provide with inspiration for their own course exercises with our extensive example repertoire. We teach UML as close to the standard as possible and illustrate all concepts using intuitive examples. The book is complemented by a website, which contains a complete set of slides to teach the contents of the book as well as teaching videos and e-learning material (http://www.uml.ac.at/).

Software modeling is a very young field of computer science. It experienced an incredible growth within the last two decades. Today, the usage of models goes far beyond pure documentation. Techniques from the area of modeling continually replace conventional programming. Models are far more than just pictures, and modeling is far more than just drawing. With our book UML@Classroom, we want to provide a solid foundation and deep understanding of the most important object-oriented modeling concepts. We aim for rising interest and enthusiasm for this exciting and extremely important field of computer science. UML@Classroom is a textbook, which explicitly addresses beginners and people with little or no modeling experience. It introduces basic concepts in a very precise manner, while abstaining from the interpretation of rare special cases. UML@Classroom is kept very compact in order to allow the reader to focus on the most commonly used concepts of UML. Much emphasis is spent on illustrative examples breathing life into the theory we present.

Acknowledgments

We would like to thank the many people who supported us in the successful completion of this book. Very special thanks go to our families who showed great patience for this book project. We are deeply indebted to Katja Hildebrandt (Vienna University of Technology) for drawing all the figures of this book and for supporting us with a million of other things. We would like to thank Ralf Gerstner from Springer and Christa Preisendanz from dpunkt.verlag for making this English version possible. Further, we would like to thank Tracey Duffy for the good collaboration on the translation of the German version of this book into English and Jeremy Gibbons (University of Oxford) for the careful proofreading and the very valuable feedback. Finally, we would like to acknowledge the input we got from our students over the years which was one of the main motivators for writing this book.

Linz and Vienna, September 2014

Martina Seidl
Marion Scholz
Christian Huemer
Gerti Kappel

Contents

1 Introduction 1
- 1.1 Motivation 1
- 1.2 Models 2
- 1.3 Object Orientation 6
  - 1.3.1 Classes 6
  - 1.3.2 Objects 6
  - 1.3.3 Encapsulation 7
  - 1.3.4 Messages 7
  - 1.3.5 Inheritance 7
  - 1.3.6 Polymorphism 8
- 1.4 The Structure of the Book 9

2 A Short Tour of UML 11
- 2.1 The History of UML 11
- 2.2 Usage 14
- 2.3 Diagrams 15
  - 2.3.1 Structure Diagrams 17
  - 2.3.2 Behavior Diagrams 19
- 2.4 Diagrams Presented in this Book 21

3 The Use Case Diagram 23
- 3.1 Use Cases 24
- 3.2 Actors 25
- 3.3 Associations 27
- 3.4 Relationships between Actors 28
- 3.5 Relationships between Use Cases 30
- 3.6 Examples of Relationships 33
- 3.7 Creating a Use Case Diagram 34
  - 3.7.1 Identifying Actors and Use Cases 34
  - 3.7.2 Describing Use Cases 35

3.7.3 Pitfalls 37
3.7.4 A Final Example 42
3.8 Summary 46

4 The Class Diagram 49
4.1 Objects 50
4.2 Classes 52
4.2.1 Notation 53
4.2.2 Attributes 54
4.2.3 Multiplicities 55
4.2.4 Operations 56
4.2.5 Visibility Markers 58
4.2.6 Class Variables and Class Operations 59
4.3 Associations 60
4.3.1 Binary Associations 60
4.3.2 N-Ary Associations 64
4.4 Association Classes 65
4.5 Aggregations 67
4.5.1 Shared Aggregations 68
4.5.2 Compositions 68
4.6 Generalizations 69
4.6.1 Inheritance 70
4.6.2 Classification 72
4.7 Abstract Classes vs. Interfaces 72
4.8 Data Types 75
4.9 Creating a Class Diagram 76
4.9.1 Generalizations 78
4.9.2 Associations and Aggregations 78
4.10 Code Generation 80

5 The State Machine Diagram 85
5.1 States and State Transitions 86
5.2 Types of States 89
5.3 Types of State Transitions 92
5.4 Types of Events 94
5.5 Composite States 96
5.5.1 The Orthogonal State 97
5.5.2 Submachines 98
5.5.3 Entry and Exit Points 99
5.5.4 The History State 101
5.6 Sequence of Events 102
5.7 Summary 105

6 The Sequence Diagram 107
6.1 Interaction Partners 108
6.2 Exchanging Messages 110
6.3 Messages 112
6.4 Combined Fragments 115
6.4.1 Branches and Loops 116
6.4.2 Concurrency and Order 119
6.4.3 Filters and Assertions 125
6.5 Further Language Elements 126
6.5.1 Interaction References 127
6.5.2 Gates 128
6.5.3 Continuation Markers 129
6.5.4 Parameters and Local Attributes 129
6.5.5 Time Constraints 130
6.5.6 State Invariants 132
6.6 Creating a Sequence Diagram 133
6.6.1 The Connection between a Class Diagram and a Sequence Diagram 133
6.6.2 Describing Design Patterns 135
6.7 The Communication, Timing, and Interaction Overview Diagrams 136
6.8 Summary 139

7 The Activity Diagram 141
7.1 Activities 142
7.2 Actions 143
7.2.1 Event-Based Actions 144
7.2.2 Call Behavior Actions 145
7.3 Control Flows 146
7.4 Object Flows 154
7.5 Partitions 156
7.6 Exception Handling 159
7.7 Concluding Example 161

8 All Together Now 167
8.1 Example 1: Coffee Machine 167
8.2 Example 2: Submission System 171
8.3 Example 3: Data Type Stack 180
8.4 Summary 182

9 Further Topics 183
9.1 Structuring Models 183
9.1.1 Packages 184
9.1.2 Importing Elements/Packages 185

9.2 The UML Metamodel 186
9.3 UML Extension Mechanisms 187
9.3.1 Stereotypes and Profiles 189
9.3.2 Applying Stereotypes of a Profile 191
9.4 Model-Based Software Development 192

References 195

Index 199