1. Record Nr.	UNINA9910254818603321
Autore	Rumpe Bernhard
Titolo	Agile Modeling with UML : Code Generation, Testing, Refactoring / / by Bernhard Rumpe
Pubbl/distr/stampa	Cham : , : Springer International Publishing : , : Imprint : Springer, , 2017
ISBN	3-319-58862-1
Edizione	[1st ed. 2017.]
Descrizione fisica	1 online resource (XIII, 388 p. 176 illus., 101 illus. in color.)
Disciplina	005.117
Soggetti	Software engineering Management information systems Computer science Software Engineering Software Management Management of Computing and Information Systems
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	1 Introduction -- 2 Agile and UML-Based Methodology -- 3 Compact Overview of UML/P -- 4 Principles of Code Generation -- 5 Transformations for Code Generation -- 6 Principles of Testing with Models -- 7 Model-Based Tests -- 8 Design Patterns for Testing -- 9 Refactoring as a Model Transformation -- 10 Refactoring of Models -- 11 Summary, Further Reading and Outlook.
Sommario/riassunto	This book focuses on the methodological treatment of UML/P and addresses three core topics of model-based software development: code generation, the systematic testing of programs using a model-based definition of test cases, and the evolutionary refactoring and transformation of models. For each of these topics, it first details the foundational concepts and techniques, and then presents their application with UML/P. This separation between basic principles and applications makes the content more accessible and allows the reader to transfer this knowledge directly to other model-based approaches and languages. After an introduction to the book and its primary goals in Chapter 1, Chapter 2 outlines an agile UML-based approach using

UML/P as the primary development language for creating executable models, generating code from the models, designing test cases, and planning iterative evolution through refactoring. In the interest of completeness, Chapter 3 provides a brief summary of UML/P, which is used throughout the book. Next, Chapters 4 and 5 discuss core techniques for code generation, addressing the architecture of a code generator and methods for controlling it, as well as the suitability of UML/P notations for test or product code. Chapters 6 and 7 then discuss general concepts for testing software as well as the special features which arise due to the use of UML/P. Chapter 8 details test patterns to show how to use UML/P diagrams to define test cases and emphasizes in particular the use of functional tests for distributed and concurrent software systems. In closing, Chapters 9 and 10 examine techniques for transforming models and code and thus provide a solid foundation for refactoring as a type of transformation that preserves semantics. Overall, this book will be of great benefit for practical software development, for academic training in the field of Software Engineering, and for research in the area of model-based software development. Practitioners will learn how to use modern model-based techniques to improve the production of code and thus significantly increase quality. Students will find both important scientific basics as well as direct applications of the techniques presented. And last but not least, the book will offer scientists a comprehensive overview of the current state of development in the three core topics it covers.