Use Case Driven Object Modeling With UML Theory and Practice 2nd Edition

Use Case Driven Object Modeling with UML Theory and Practice shows how to drive an object-oriented software design from use case all the way through coding and testing, based on the minimalist, UML-based ICONIX process. In addition to a comprehensive explanation of the foundations of the approach, the book makes extensive use of examples and provides exercises at the back of each chapter.

Introduction to ICONIX Process· Domain Modeling· Use Case Modeling· Requirements Review· Robustness Analysis· Preliminary Design Review· Technical Architecture· Sequence Diagrams· Critical Design Review· Implementation: Getting from Detailed Design to Code· Code Review and Model Update· Design-Driven Testing· Addressing Requirements

Overview of the ICONIX process for designing and developing software systems. ICONIX has developed a growing following over the years. Object Modeling with UML - Theory and Practice combines the notation of UML with a lightweight but effective process - the ICONIX process - for designing and developing software systems. ICONIX offers just enough structure to be successful.

This is the fourth report on mothers and babies in NSW to combine the annual reports of the NSW Midwives Data Collection (MDC), the Neonatal Intensive Care Units' Data Collection and the NSW Birth Defects Register."

Ground-Truthing, Programming, Formulating Topological UML Modeling Agile Model-Driven Development with UML 2.0


Offers comprehensive coverage of all major modeling viewpoints Provides details of collaboration and class diagrams for filling in the design-level models

More than 300,000 developers have benefited from past editions of UML Distilled. This third edition is the best resource for quick, no-nonsense insights into understanding and using UML 2.0 and prior versions of the UML. Some readers will want to quickly get up to speed with the UML 2.0 and learn the essentials of the UML. Others will use this book as a handy, quick reference to the most common parts of the UML. The author delivers on both of these promises in a short, concise, and focused presentation. This book describes all the major UML diagram types, what they're used for, and the basic notation involved in creating and deciphering them. These diagrams include class, sequence, object, package, deployment, use case, state machine, activity, communication, composite structure, component, interaction overview, and timing diagrams. The examples are clear and the explanations cut to the fundamental design logic. Includes a quick reference to the most useful parts of the UML notation and a useful summary of diagram types that were added to the UML 2.0. If you are like most developers, you don't have time to keep up with all the new innovations in software engineering. This new edition of Fowler's classic work gets you acquainted with some of the best thinking about efficient object-oriented software design using the UML—in a convenient format that will be essential to anyone who designs software professionally.

For courses in Software Engineering, Software Development, or Object-Oriented Design and Analysis at the Junior/Senior or Graduate level. This text can also be utilized in short technical courses or in short, intensive management courses. Shows students how to use both the principles of software engineering and the practices of various object-oriented tools, processes, and products. Using a step-by-step case study to illustrate the concepts and topics in each chapter, Bruegge and Dutoit emphasize learning object-oriented software engineer through practical experience: students can apply the techniques learned in class by implementing a real-world software project. The third edition addresses new trends, in particular agile project management (Chapter 14 Project M anagement) and agile methodologies (Chapter 16 M et hodologies).

Introduces CRC (Class, Responsibility, Collaborator) cards and describes how they can be used in interactive sessions to develop an object-oriented model of an application.

Model Driven Architecture with Executable UML Use Case Driven Object Modeling with UML: Theory And Practice Object-oriented Design and Architecture Use Case Modeling UML, Use Cases, Patterns, and Software Architectures Using the Unified Modeling Language The Case Against XP

Diagramming and process are important topics in today's software development world, as the UML diagramming language has come to be almost universally accepted. Yet process is necessary; by themselves, diagrams are of little use. Use Case Driven Object Modeling with UML - Theory and Practice combines the notation of UML with a lightweight but effective process - the ICONIX process - for designing and developing software systems. ICONIX has developed a growing following over the years. Sitting between the free-for-all of Extreme Programming and overly rigid processes such as RUP, ICONIX offers just enough structure to be successful.
This compact book helps application developers bridge the gap between the theory of the newly created Unified Software Development Process and the practical realities necessary to design and build a software system. The authors present the key ingredients of the Unified Process and demonstrate how the process was conceived to work with UML, emphasizing the application of Use Cases as a primary design tool. The book incorporates a wealth of practical experience showcased by four case studies -- a hospital information system, a video on demand system, a portfolio management system, and a vehicle navigation (IVHS) system.

A laboratory study that investigates how algorithms come into existence. Algorithms--often associated with the terms big data, machine learning, or artificial intelligence--underlie the technologies we use every day, and disputes over the consequences, actual or potential, of new algorithms arise regularly. In this book, Florian Jaton offers a new way to study computerized methods, providing an account of where algorithms come from and how they are constituted, investigating the practical activities by which algorithms are progressively assembled rather than what they may suggest or require once they are assembled.

Introduces the Unified Modeling Language, explains the fundamentals of modeling elements, structures, and the behaviors of object-oriented software systems, and offers real-world examples.

- Fast Track UML 2.0
- Modelling Foundations and Applications
- Iconix Process Roadmaps
- People, Process, and Pragmatism
- Advances and Applications in Model-Driven Engineering
- Flow
- Using CRC Cards

Neil Gaiman's perennial favorite, The Graveyard Book, has sold more than one million copies and is the only novel to win both the Newbery Medal and the Carnegie Medal. Bod is an unusual boy who inhabits an unusual place—he's the only living resident of a graveyard. Raised from infancy by the ghosts, werewolves, and other cemetery denizens, Bod has learned the antiquated customs of his guardians' time as well as their ghostly teachings—such as the ability to Fade so mere mortals cannot see him. Can a boy raised by ghosts face the wonders and terrors of the worlds of both the living and the dead? The Graveyard Book is the winner of the Newbery Medal, the Carnegie Medal, the Hugo Award for best novel, the Locus Award for Young Adult novel, the American Bookseller Association's "Best Indie Young Adult Buzz Book," a Horn Book Honor, and Audio Book of the Year.

This book offers a unique insight into a revolution in software development that allows model specifications to be fully and efficiently translated into code. Using the most widely adopted, industry standard, software modelling language, UML, the reader will learn how to build robust specifications based on OMG's Model Driven Architecture (MDA). From there, the authors describe the steps needed to translate the Executable UML (xUML) models to any platform-specific implementation. The benefits of this approach go well beyond simply reducing or eliminating the coding stage - it also ensures platform independence, avoids obsolescence (programming languages may change, the model doesn't) and allows full verification of the models by executing them in a test and debug xUML environment. This is an excellent reference for anyone embarking on what is surely the future of software development for medium and large scale projects.

Larman covers how to investigate requirements, create solutions and then translate designs into code, showing developers how to make practical use of the most significant recent developments. A summary of UML notation is included

- The Constitution of Algorithms
- The Unified Software Development Process
- Applying Use Case Driven Object Modeling with UML
- Working Smart to Deliver Quality
- Executable UML
- The Art of Modeling Software Systems Demonstrated through Worked Examples and Solutions
- Use Case Driven Object Modeling with UML

Based on Objectory which is the first commercially available comprehensive object-oriented process for developing large scale industrial systems.

This book covers all you need to know to model and design software applications from use cases to software architectures in UML and shows how to apply the COMET UML-based modeling and design method to real-world problems. The author describes architectural patterns for various architectures, such as broker, discovery, and transaction patterns for service-oriented architectures, and addresses software quality attributes including maintainability, modifiability, testability, traceability, scalability, reusability, performance, availability, and security. Complete case studies illustrate design issues for different software architectures: a banking system for client/server architecture, an online shopping system for service-oriented architecture, an emergency monitoring system for component-based software architecture, and an automated guided vehicle for real-time software architecture. Organized as an introduction followed by several short, self-contained chapters, the book is perfect for senior undergraduate or graduate courses in software engineering and design, and for experienced software engineers wanting a quick reference at each stage of the analysis, design, and development of large-scale software systems.

Topological UML Modeling: An Improved Approach for Domain Modeling and Software Development presents a specification for Topological UML® that combines the formalism of the Topological Functioning Model (TFM) mathematical topology with a
specified software analysis and design method. The analysis of problem domain and design of desired solutions within software development processes has a major impact on the achieved result of developed software. While there are many tools and different techniques to create detailed specifications of the solution, the proper analysis of problem domain functioning is ignored or covered insufficiently. The design of object-oriented software has been led for many years by the Unified Modeling Language (UML®), an approved industry standard modeling notation for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system, and this comprehensive book shines new light on the many advances in the field. Presents an approach to formally define, analyze, and verify functionality of existing processes and desired processes to track incomplete or incorrect functional requirements Describes the path from functional and nonfunctional requirements specification to software design with step-by-step creation and transformation of diagrams and models with very early capturing of security requirements for software systems. Defines all modeling constructs as extensions to UML®, thus creating a new UML® profile which can be implemented in existing UML® modeling tools and toolsets This book constitutes the proceedings of the 6th European Conference on Modelling Foundations and Applications, held in Paris, France, in June 2010.

**Theory and Practice**

**Software Modeling and Design**

**UML Explained**

An Improved Approach for Domain Modeling and Software Development

**Visual Modeling with Rational Rose 2000 and UML**

**APPLYING UML & PATTERNS 3RD EDITION**

**A Brief Guide to the Standard Object Modeling Language**

Broadly-scoped requirements such as security, privacy, and response time are a major source of complexity in modern software systems. This is due to their tangled inter-relationships with and effects on other requirements. Apect-Oriented Requirements Engineering (AORE) aims to facilitate modularisation of such broadly-scoped requirements, so that software developers are able to reason about them in isolation - one at a time. AORE also captures these inter-relationships and effects in well-defined composition specifications, and, in so doing exposes the causes for potential conflicts, trade-offs, and roots for the key early architectural decisions. Over the last decade, significant work has been carried out in the field of AORE. With this book the editors aim to provide a consolidated overview of these efforts and results. The individual contributions discuss how aspects can be identified, represented, composed and reasoned about, as well as how they are used in specific domains and in industry. Thus, the book does not present one particular AORE approach, but conveys a broad understanding of the aspect-oriented perspective on requirements engineering. The chapters are organized into five sections: concern identification in requirements, concern modelling and composition, domain-specific use of AORE, aspect interactions, and AORE in industry. This book provides readers with the most comprehensive coverage of AORE and the capabilities it offers to those grappling with the complexity arising from broadly-scoped requirements - a phenomenon that is, without doubt, universal across software systems. Software engineers and related professionals in industry, as well as advanced undergraduate and post-graduate students and researchers, will benefit from these comprehensive descriptions and the industrial case studies.

**Use Case Driven Object Modeling with UML Theory and Practice**

Apress

This book discusses how model-based approaches can improve the daily practice of software professionals. This is known as Model-Driven Software Engineering (MDSE) or, simply, Model-Driven Engineering (MDE). MDSE practices have proved to increase efficiency and effectiveness in software development, as demonstrated by various quantitative and qualitative studies. MDSE adoption in the software industry is foreseen to grow exponentially in the near future, e.g., due to the convergence of software development and business analysis. The aim of this book is to provide you with an agile and flexible tool to introduce you to the MDSE world, thus allowing you to quickly understand its basic principles and techniques and to choose the right set of MDSE instruments for your needs so that you can start to benefit from MDSE right away. The book is organized into two main parts. The first part discusses the foundations of MDSE in terms of basic concepts (i.e., models and transformations), driving principles, application scenarios, and current standards, like the well-known MDA initiative proposed by OMG (Object Management Group) as well as the practices on how to integrate MDSE in existing development processes. The second part deals with the technical aspects of MDSE, spanning from the basics on when and how to build a domain-specific modeling language, to the description of M/odels-to-Text and M/odels-to-M/odels transformations, and the tools that support the management of MDSE projects. The second edition of the book features: a set of completely new topics, including: full example of the creation of a new modeling language (IFML), discussion of modeling issues and approaches in specific domains, like business process modeling, user interaction modeling, and enterprise architecture complete revision of examples, figures, and text, for improving readability, understandability, and coherence better formulation of definitions, dependencies between concepts and ideas addition of a complete index of book content In addition to the contents of the book, more resources are provided on the book's website http://www.mdse-book.com, including the examples presented in the book. Discusses how to define and organize use cases that model the user requirements of a software application. The approach focuses on identifying all the parties who will be using the system, then writing detailed use case descriptions and structuring the use case model. An ATM example runs throughout the book. The authors work at Rational Software. A notation copyrighted by Book News, Inc., Portland, OR

Succeeding with Use Cases

Model-Driven Software Engineering in Practice

Design Driven Testing

Extreme Programming Refactored

Step-By-Step Guidance for Soa, Embedded, and Algorithm-Intensive Systems

Apect-Oriented Requirements Engineering
UML in Practice

The groundbreaking book Design Driven Testing brings sanity back to the software development process by flipping around the concept of Test-Driven Development (TDD)—restoring the concept of using testing to verify a design instead of pretending that unit tests are a replacement for design. Anyone who doubts that TDD is “Too Damn Difficult” will appreciate this book. Design Driven Testing shows that, by combining a forward-thinking development process with cutting-edge automation, testing can be a finely targeted, business-driven, rewarding effort. In other words, you’ll learn how to test smarter, not harder. Applies a feedback-driven approach to each stage of the project lifecycle. Illustrates a lightweight and effective approach using a core subset of UML. Follows a real-life example project using Java and Flex/ActionScript. Presents bonus chapters for advanced DD'Ters covering unit-test antipatterns (and their opposite, “test-conscious” design patterns), and showing how to create your own test transformation templates in Enterprise Architect. ICONIX Process has a long track record of helping companies avoid analysis paralysis on a multitude of projects, and is best suited for developing Web and GUI-based systems. This resource contains a treasure-trove of tailored roadmaps, proven on demanding real-life projects.

Venturing beyond C++ programming, this text shows how to engineer software products using object-oriented principles. It covers gathering requirements, specifying objects, object verification, defining relations between objects, translating object design into code, object testing, and software maintenance.

From best-selling author and noted teacher and speaker Yehuda Berg comes a thought-provoking call to action on our current global crisis. Positing that our collective abdication of responsibility — in every facet of our lives, including business and the economy, the environment, government and politics, healthcare, education, and religion — has contributed to the problems and challenges we face, Berg asserts that taking responsibility for our actions (or lack thereof) and their consequences is the key to achieving change for the better. Berg urges readers to access the power within each of us, using the principles of Kabbalah, in order to create the consciousness shift required for lasting positive change.

Use Case Driven Object Modeling with UML Theory and Practice

Annotated E-commerce Example

UML Distilled

Object-Oriented Systems Development

Object-Oriented Modeling and Design with UML

Object-Oriented Software Modeling Using UML, Patterns, and Java: Pearson New International Edition

The Graveyard Book

As organizations and research institutions continue to emphasize model-driven engineering (MDE) as a first-class approach in the software development process, the utilization of software in multiple domains and professional networks is becoming increasingly vital. Advances and Applications in Model-Driven Engineering explores this relatively new approach in software development that can increase the level of abstraction of development of tasks. This publication covers the issues of bridging the gaps between various disciplines within software engineering and computer science. Professionals, researchers, and students will discover the most current tools and techniques available in the field to maximize efficiency of model-driven software development.

The first UML book to cover Rational Rose 2000, this brand-new edition reviews the three key interrelated components of state-of-the-art software system design: the Rational Unified process, the Unified Modeling Language, and Rational Rose 2000. Then, through a simplified case study, it walks developers through a real-world business system. Includes screen shots demonstrating UML at work in the Rational Rose 2000 modeling tool.

The acclaimed beginner’s book on object technology now presents UML 2.0, Agile Modeling, and the latest in object development techniques.

THE BESTSELLING CLASSIC ON "FLOW" — THE KEY TO UNLOCKING MEANING, CREATIVITY, PEAK PERFORMANCE, AND TRUE HAPPINESS

Legendary psychologist Mihaly Csikszentmihalyi’s famous investigations of “optimal experience” have revealed that what makes an experience genuinely satisfying is a state of consciousness called flow. During flow, people typically experience deep enjoyment, creativity, and a total involvement with life. In this new edition of his groundbreaking classic work, Csikszentmihalyi (the leading researcher into ‘flow states’” — Newweek) demonstrates the ways this positive state can be controlled, not just left to chance. Flow: The Psychology of Optimal Experience teaches how, by ordering the information that enters our consciousness, we can discover true happiness, unlock our potential, and greatly improve the quality of our lives. "Explores a happy state of mind called flow, the feeling of complete engagement in a creative or playful activity." — Time

A Foundation for Model-driven Architecture

A Use Case Driven Approach

An Informal Approach to Object-Oriented Development

The Object Primer

Agile Development with ICONIX Process

The Psychology of Optimal Experience

From the beginning of software time, people have wondered why it isn’t possible to accelerate software projects by simply adding staff. This is sometimes known as the “nine women can’t make a baby in one month” problem. The most famous treatise declaring this to be impossible is Fred Brooks’ 1975 book The Mythical Man-Month, in which he declares that adding more programmers to a late software project makes it later. And indeed this has proven largely true over the decades. Aided by a domain-driven code generator that quickly creates database and API code, Parallel Agile (PA) achieves significant schedule compression using parallelism: as many developers as necessary can independently and concurrently develop the scenarios from initial prototype through production code. Projects can scale by elastic staffing, rather than by stretching schedules for larger development efforts. Schedule compression with a large team of developers working in parallel is analogous to hardware acceleration of compute problems using parallel CPUs. PA has some similarities with and differences from other Agile approaches. Like most Agile methods, PA “gets to code early” and uses feedback from executable software to drive requirements and design. PA uses technical prototyping as a risk-mitigation strategy, to help sanity-check requirements for feasibility, and to evaluate different technical architectures and technologies. Unlike many Agile methods, PA does not support “design by refactoring,” and it doesn’t drive designs from unit tests. Instead, PA uses a minimalist UML-based design approach (Agile/ICONIX) that starts out with a domain model to facilitate communication across the development team, and partitions the system along use case boundaries, which enables parallel development. Parallel Agile is fully compatible with the Incremental Commitment Spiral Model (ICSM), which involves concurrent effort of a systems engineering team, a development team, and a test team working alongside the developers. The authors have been researching and refining the PA process for several years on multiple test projects that have involved over 200 developers. The book’s example project details the design of one of these test projects, a crowdsourced traffic safety system.

* Examples are easy to understand; diagrams aren't overly busy. * Written in user-friendly style author is known for. * Condensed, distilled presentation of the UML Superstructure document will get you up to speed with UML 2.0.

This is the eBook of the printed book and may not include any media, website access codes, or print supplements that may come packaged with the bound book. This revision offers a crisp, clear explanation of the basics of object-oriented thinking via UML models, then presents a process for applying these
principles to software development, including C++, Java, and relational databases. An integrated case study threads throughout the book, illustrating key ideas as well as their application.

Gain the skills to effectively plan software applications and systems using the latest version of UML. UML 2 represents a significant update to the UML specification, from providing more robust mechanisms for modeling workflow and actions to making the modeling language more executable. Now in its second edition, this bestselling book provides you with all the tools you need for effective modeling with UML 2. The authors get you up to speed by presenting an overview of UML and its main features. You'll then learn how to apply UML to produce effective diagrams as you progress through more advanced topics such as use-case diagrams, classes and their relationships, dynamic diagrams, system architecture, and extending UML. The authors take you through the process of modeling with UML so that you can successfully deliver a software product or information management system. With the help of numerous examples and an extensive case study, this book teaches you how to:

* Organize, describe, assess, test, and realize use cases
* Gain substantial information about a system by using classes
* Utilize activity diagrams, state machines, and interaction diagrams to handle common issues
* Extend UML features for specific environment or domains
* Use UML as part of a Model Driven Architecture initiative
* Apply an effective process for using UML

The CD-ROM contains all of the UML models and Java™ code for a complete application, Java™ 2 Platform, Standard Edition, Version 1.4.1, and links to the Web sites for vendors of UML 2 tools.

Parallel Agile – faster delivery, fewer defects, lower cost

UML 2 Toolkit
A Practical Approach
An Annotated E-commerce Example
Second Edition

The Art of Objects offers an extensive overview of the long-standing principles of object technology, along with leading-edge developments in the field. It will give you a greater understanding of design patterns and the know-how to use them to find effective solutions to a wide range of design challenges. And because the book maintains an approach independent of specific programming languages, the concepts and techniques presented here can be applied to any object-oriented development environment. Using the Unified Modeling Language (UML), The Art of Objects examines numerous static and dynamic practical object design patterns, illustrated by real-life case studies that demonstrate how to put the patterns to work. You will also find discussion of basic concepts of database management and persistent objects, and an introduction to advanced topics in object modeling and interface design patterns. Moving beyond the design level, the book also covers important concepts in object-oriented architecture. Specific topics include:

* Object creation and destruction, associations and links, aggregation, inheritance, and other object design fundamentals
* UML notation basics for static and dynamic

Copyright: app.inquivix.com