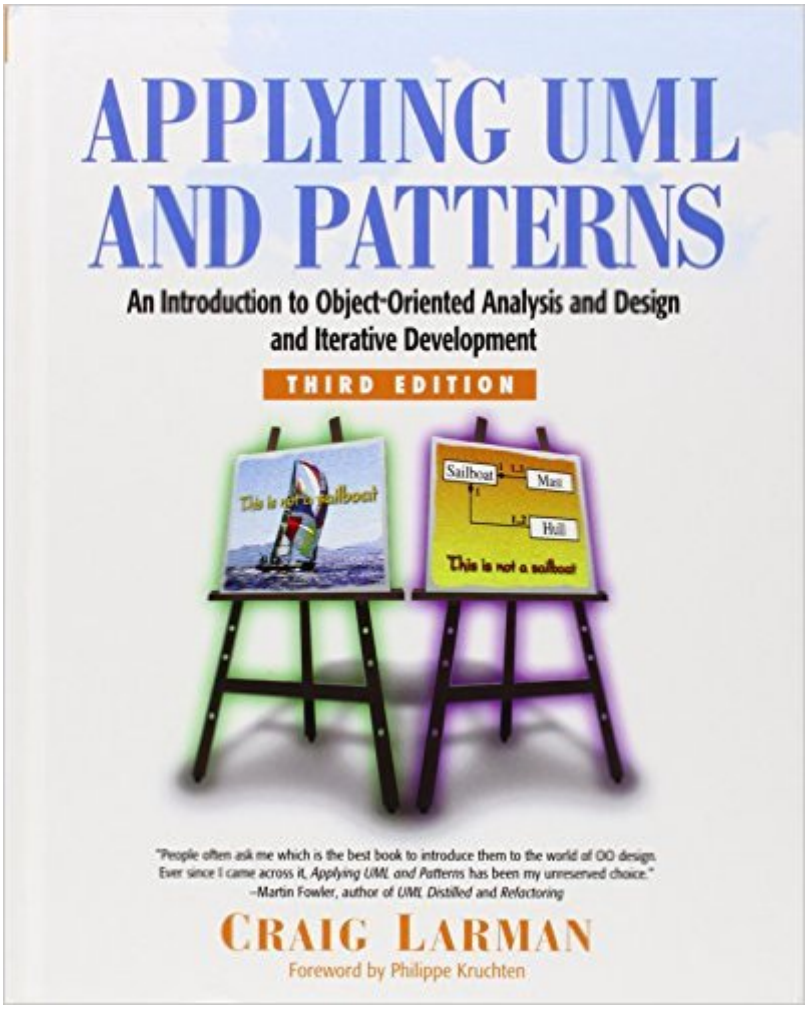
Applying UML And Patterns: An Introduction To Object-Oriented Analysis And Design And Iterative Development (3rd Edition)



Synopsis

â œThis edition contains Larmanâ ™s usual accurate and thoughtful writing. It is a very good book made even better.â •â ”Alistair Cockburn, author, Writing Effective Use Cases and Surviving OO Projects â œToo few people have a knack for explaining things. Fewer still have a handle on software analysis and design. Craig Larman has both.â •â ”John Vlissides, author, Design Patterns and Pattern Hatching â œPeople often ask me which is the best book to introduce them to the world of OO design. Ever since I came across it Applying UML and Patterns has been my unreserved choice.â •â ”Martin Fowler, author, UML Distilled and Refactoring â œThis book makes learning UML enjoyable and pragmatic by incrementally introducing it as an intuitive language for specifying the artifacts of object analysis and design. It is a well written introduction to UML and object methods by an expert practitioner.â •â ”Cris Kobryn, Chair of the UML Revision Task Force and UML 2.0 Working Group A brand new edition of the worldâ ™s most admired introduction to object-oriented analysis and design with UML Fully updated for UML 2 and the latest iterative/agile practices Includes an all-new case study illustrating many of the bookâ ™s key points Applying UML and Patterns is the worldâ ™s #1 business and college introduction to â œthinking in objectsâ •â ”and using that insight in real-world object-oriented analysis and design. Building on two widely acclaimed previous editions, Craig Larman has updated this book to fully reflect the new UML 2 standard, to help you master the art of object design, and to promote high-impact, iterative, and skillful agile modeling practices. Developers and students will learn object-oriented analysis and design (OOA/D) through three iterations of two cohesive, start-to-finish case studies. These case studies incrementally introduce key skills, essential OO principles and patterns, UML notation, and best practices. You wonâ ™t just learn UML diagramsâ ”youâ ™ll learn how to apply UML in the context of OO software development. Drawing on his unsurpassed experience as a mentor and consultant, Larman helps you understand evolutionary requirements and use cases, domain object modeling, responsibility-driven design, essential OO design, layered architectures, â œGang of Fourâ • design patterns, GRASP, iterative methods, an agile approach to the Unified Process (UP), and much more. This editionâ ™s extensive improvements include A stronger focus on helping you master OOA/D through case studies that demonstrate key OO principles and patterns, while also applying the UML New coverage of UML 2, Agile Modeling, Test-Driven Development, and refactoring Many new tips on combining iterative and evolutionary development with OOA/D Updates for easier study, including new learning aids and graphics New college educator teaching resources Guidance on applying the UP in a light, agile spirit, complementary with other iterative methods such as XP and Scrum Techniques for applying the UML to documenting architectures A

new chapter on evolutionary requirements, and much more Applying UML and Patterns, Third Edition, is a lucid and practical introduction to thinking and designing with objectsâ ”and creating systems that are well crafted, robust, and maintainable.

Book Information

Hardcover: 736 pages

Publisher: Prentice Hall; 3 edition (October 30, 2004)

Language: English

ISBN-10: 0131489062

ISBN-13: 978-0131489066

Product Dimensions: 8.3 x 1.7 x 10.1 inches

Shipping Weight: 3.7 pounds (View shipping rates and policies)

Average Customer Review: 4.5 out of 5 starsÂ Â See all reviewsÂ (50 customer reviews)

Best Sellers Rank: #91,647 in Books (See Top 100 in Books) #8 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > UML #45 inÂ Books > Textbooks > Computer Science > Object-Oriented Software Design #158 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Object-Oriented Design

Customer Reviews

There is a lot of textbooks on UML in the market, similarly on development processes like the Unified Process, design patterns and OOA/D. Many textbooks that I have seen provide a dry list of UML notations, or a dry list of process guidelines, or trivial examples on how a design pattern can be implemented. However, no other textbook in my opinion makes an excellent job in putting everything together in a case study (the 3rd edition provides two case studies) in order to illustrate (1) what is the significance of each one of the above, (2) how they fit together and (3) what are possible tradeoffs. The author very clearly explains what are the underlying principes behind object-oriented software development and (more importantly) how these principles can be put into practice.Since the first edition I found Craig’s writing style very easy to follow and as a graduate student taking software engineering and related classes I used this textbook as a self study to learn about OOA/D and UML. As an instructor I have been using this textbook for a number of software engineering and related classes (both senior level undergraduate and graduate), and the feedback I receive from students is very positive. I also recommend this book to students who are undertaking final-year undergraduate projects or graduate projects, and we have found this book to be very valuable for projects that involve several stages of analysis, design and implementation and who

want to know how a process such as the Unified Process can be used in an agile manner. My experience tells me that this last point is very important for students who would work individually or in small groups over a (usually) short period of time to complete a development project.Several of my previous students who are now employed in the IT industry as developers are telling me that they still use this book and find it a very valuable reference.The book has also sparked interesting discussions among colleagues and researchers on various aspects on OOA/D and it is a valuable source. More particularly, the book successfully manages to integrate the principle of Design by Contract beyond implementation. Craig's approach to introduce operation contracts places emphasis on assertions from early stages of development and shows how this emphasis is propagated to detailed design (through UML communication diagrams) and through the use of responsibility patterns.Regarding a comment on GRASP by a previous (and anonymous) reviewer, I would like to point out that a pattern is a set of principles (can be on any level of granularity) that solves a recurring problem at any stage during development. This (albeit informal) definition does not confine patterns to structural or behavioral design (along the lines of the GoF design patterns). Craig makes that very clear in the book particularly in the second and third edition) and I'm afraid to say that the reviewer who made the comment either skipped that part or misunderstood it.

One of the more difficult concepts to bring to programming is the very basic concept of Object Orientation. Most programming efforts in the college/university level are really short and quick, while most software projects in the real world are much bigger. Combining all this together you have the potential for turning out graduates that have a hard time in the real world.An interesting point of this book is its overall design, which is laid out like a software project. That way you are working within the broad concepts while you don't even know that you are being exposed to them.This is not a book on programming. You should know at least one object oriented language before beginning it. Java is used for most examples, but one of the C's or Python could be used.The title of the book is somewhat misleading to me. True it is about UML and Patterns, but it's really the sub-title that tells the story. This is a book on object oriented analysis and design (OOAD). UML and Patterns are simply two of the tools used to teach OOAD.

Let me say to begin that I am a graduate student in computer engineering, without a strong OO background. Sure I knew inheritance, polymorphism, and even some UML. But how do you really use them in practice? I have been eager to learn what this OOAD is all about, and anyway it's a valuable skill to possess.Now where to begin learning OOAD? As I scratched the surface I

encountered such oft-cited works asÂ Design PatternsÂ by the "Gang of Four", Booch’sÂ Object-Oriented Analysis and Design with Applications, andÂ Object-Oriented Modeling and DesignÂ by Rumbaugh et al. Obviously many books attempt to explain the OO paradigm. Specifically I want one that is: 1.) interesting, 2.) informed, and 3.) insightful.That’s why I’m glad I chose this book. It’s unmistakably for serious readers, and not as easily accessible or "witty" as a few others. On the other hand, if you want to encounter *many* useful concepts and suggestions from an authoritative source, then I can’t imagine a better choice than Applying UML and Patterns. I’ve read it cover-to-cover once, and have already begun referring back to it for my own purposes.Sometimes it’s useful to understand the author’s perspective, to know if you will learn anything useful from their books. Craig Larman is obviously a proponent of agile risk-driven software development, OOAD, and using the UML sparsely as a communicative tool ("sketching" vs. "blueprinting").Larman makes a very strong case for his perspective, too. After all, everyone knows requirements evolve and change over time, as does design. So why not adopt a process that accomodates this? Similarly, the UML is potentially a complicated language, but why get caught up on notation? The point is to communicate something of value, especially during design, when collaborative decisions must be made - leave the rest to CASE tools. Don’t be scared of the Unified Process either, as it provides a great context in which to discuss business processes and risk-driven software development, even if you never explicitly use it.By the time you finish this book you will: have a good overview of iterative and agile software development, know aspects of the unified process, know the basics of the OO paradigm, know how to assign responsibilities to objects, have been exposed to the most common design patterns, have encountered a few analysis patterns, and have a wealth of tips and suggestions to draw from in your own work.All of this is presented in the context of a case study on a fictional point-of-sale system. The book slightly favors Java in its examples, but as these are fairly sparse and generally brief, it should be easy enough to follow for those familiar with C++ or C#. The author tries to note whenever choice of language has a significant impact.Even at nearly 40 chapters, I wish the book were longer, as Larman’s writing style is coherent and enjoyable. You’ll likely find yourself wanting to know more about software architecture or the details of certain patterns, and luckily the book is full of citations and suggested reading material.It’s a great place to start for students and professionals, anyone who wants to pick up OOAD. If you only want a reference on patterns, then this is probably not the book for you. It doesn’t go into great detail about the more complex patterns. Therefor it’s recommended that you own some of the classic patterns literature. Likewise if you primarily need a reference on UML, I’d recommend Martin Fowler’s excellentÂ UML Distilled. Again, the bibliography of Applying UML and

Patterns is an abundant source of related works, for those digging a bit deeper.

Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development (3rd Edition) Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process (2nd Edition) Object Success : A Manager's Guide to Object-Oriented Technology And Its Impact On the Corporation (Object-Oriented Series) Object-Oriented Software Engineering Using UML, Patterns, and Java (3rd Edition) [Economy Edition] Object-Oriented Analysis and Design for Information Systems: Modeling with UML, OCL, and IFML UML and the Unified Process: practical object-oriented analysis and design Systems Analysis and Design: An Object-Oriented Approach with UML Systems Analysis and Design with UML Version 2.0: An Object-Oriented Approach Reusable Software : The Base Object-Oriented Component Libraries (Prentice Hall Object-Oriented Series) Object-Oriented Software Engineering: Using UML, Patterns and Java (2nd Edition) The Object-Oriented Approach: Concepts, Systems Development, and Modeling with UML, Second Edition Object-Oriented Software Engineering: Practical Software Development Using UML and Java Object-Oriented Modeling and Design with UML (2nd Edition) Fundamentals of Object-Oriented Design in UML Object-Oriented Technology: From Diagram to Code with Visual Paradigm for UML Real Time UML: Advances in the UML for Real-Time Systems (3rd Edition) Visual Object-Oriented Programming Using Delphi With CD-ROM (SIGS: Advances in Object Technology) Design Patterns CD: Elements of Reusable Object-Oriented Software (Professional Computing) Design Patterns: Elements of Reusable Object-Oriented Software Design Patterns: Elements of Reusable Object-Oriented Software (Adobe Reader)