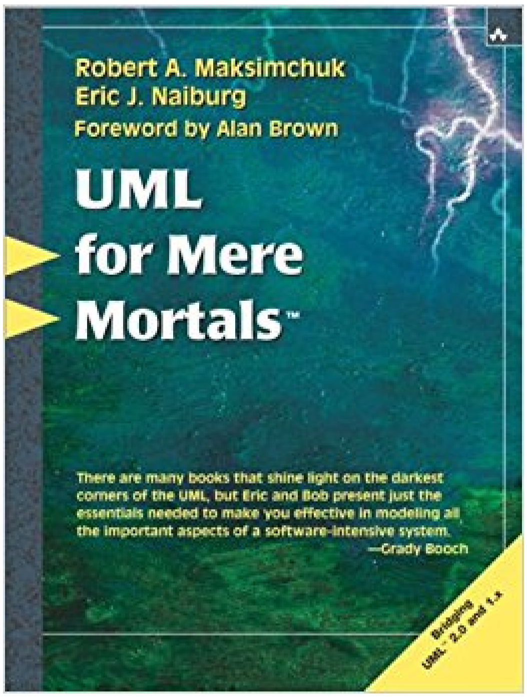
UML For Mere Mortals



Robert A. Maksimchuk
Eric J. Naiburg
Foreword by Alan Brown

UML for Mere Mortals™

There are many books that shine light on the darkest corners of the UML, but Eric and Bob present just the essentials needed to make you effective in modeling all the important aspects of a software-intensive system.
—Grady Booch

Synopsis

The Unified Modeling Language (UML) is the standard notation forarchitecting software and systems. Even though the UML has existed for years,no book has offered the fresh perspective on the industry standard that UMLfor Mere Mortals does. The book introduces the UML to those technical andnon-technical readers who are stakeholders in software development projects,but are not hard-core modelers. It does so in a manner that is easy tounderstand, addressing the core concepts of the UML that all readers need toknow to address the UML in terms of the specific concerns of thestakeholders. But above all, the book most important feature is that it ispragmatic. The authors share real world experiences and focus on what thesereaders will run into on real world projects.

Book Information

Paperback: 288 pages

Publisher: Addison-Wesley Professional; 1 edition (November 5, 2004)

Language: English

ISBN-10: 0321246241

ISBN-13: 978-0321246240

Product Dimensions: 6.8 x 0.7 x 9.2 inches

Shipping Weight: 1.2 pounds

Average Customer Review: 4.3 out of 5 starsÂ Â See all reviewsÂ (7 customer reviews)

Best Sellers Rank: #1,383,501 in Books (See Top 100 in Books) #101 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > UML #473 inÂ Books > Textbooks > Computer Science > Object-Oriented Software Design #1623 inÂ Books > Computers & Technology > Programming > Software Design, Testing & Engineering > Object-Oriented Design

Customer Reviews

The sheer number of books on UML is simply amazing, and it seems like finding a right one for you is a task all into itself. No matter whom you want to become, a hardcore UML modeler or a weekend reader feeding one’s curiosity, the book "UML for Mere Mortals" is a great way to start. The main and important topics are covered, and the details are left untold. That’s perfectly ok, since even the UML professionals don’t refer to all aspects of UML due to its complexity. Simple UML diagrams are easy to grasp, but UML for large projects get very complicated, making the users of UML stick to common diagrams in order to get the point across more easily to readers. What is the point of a complex and intertwined diagram if you are the only one that can read it?It is crucial to keep in mind that the goal is to model your enterprise in order to have a common language across all aspects of business via which everyone can communicate. What is the point of accomplishing this task if no one else in your enterprise can understand what you are trying to say? You have accomplished nothing, and only wasted away hours of work. The authors of the book have this mentality in mind when they are talking about UML. They start with basic stuff such as Business Modeling and Business Use Cases: a top-down approach if you will. The fact of the matter is that UML can readily model all aspects of an enterprise from what is called Business Use Cases all the way down to how each executable piece of software is deployed.After Business Modeling has been accomplished, it is onto requirements modeling with Use Cases. A Use Case driven process, where your capture your requirement solely using Use Cases has shown to be the best way to start a new project.

I don’t care what the experts say... UML isn’t intuitive nor is it "easy" to read. Learning to use it can be intimidating. UML For Mere Mortals by Robert A. Maksimchuk and Eric J. Naiburg is a very nice way to get your feet wet on the subject...Chapter List: Introduction to the UML; Business Models; Requirements Modeling; Architectural Modeling; Application Modeling; Database Modeling; Testing; Is That All There Is?; How Do I Get Started Using The UML?; Where Can I Learn More?; Glossary; Answers To Review Questions; UML Diagrams and Elements; IndexI’ve read a few books on UML, and it’s pretty easy to get bogged down in all the rules and minutiae. UML is one of those things that can have the experts arguing about fine distinctions that you’ll never experience in your working career. In this book, you can forget all that. The authors don’t try to teach you absolutely everything there is to know. The goal is to focus on practical usage and cover those things that you’ll most likely run up against in real life. And in my opinion, they nail that goal. Most of the subtopics within each chapter have a topic heading that is a question. The questions are ones that you’d encounter as an actual student of UML (like how do I model my business using the UML?), and that tends to make sure the subject matter stays practical and useful. There are also a number of very good sidebars that cover lessons learned, real world experience, things to watch out for, and "deep dive" items that cover things in a bit more depth. There are even review questions you can use to see how much you’ve retained. All in all, a good format and packaging of the material.This is the first "Mere Mortals" title I’ve read, and I don’t think it will be my last.

UML is still as much an art as a science, and as proof I’d offer the fact that UML - despite industry-wide agreement of it being a Very Good Idea - still has very few native "speakers" who can offer the formula for how to do it successfully. Witness the effort of the two authors with decades of previous success in applying UML (and, as members of the Rational team, much future success tied to the successful application of UML) as they dispense the following:Rule 1: "Use cases should not describe how the scenario will be implemented." - p.67 (this is the traditional design dogma, that it is even possible to describe a problem in a meaningful way that produces value without first acknowledging some aspect of the solution - see Kovitz's "Practical Software Requirements" for more on this)Example: The authors offer an example where an actor (Driver) will perform a scenario (Take Trip) which will always include another use case (Fuel Vehicle).Inconsistency:* By specifying Driver, haven't we already made some assumptions about the particular implementation - that we are describing a car as the vehicle, and that a vehicle is even necessary to take a trip? Can't the individual just walk?* Does every occurence require a vehicle to be fueled? What about a trip across town?* Could this be a bike rather than a motorized vehicle?Assessment: The point of this use case would be that the value proposition to the "driver" is to take a trip, not the act of driving a vehicle, so therefore it would make more sense to abstract the vehicle out of the equation and therefore break the inclusion of the Fuel Vehicle case.