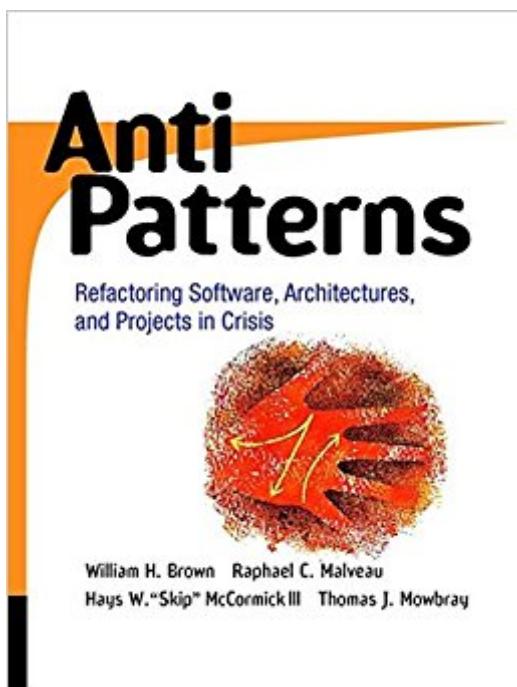The book was found

AntiPatterns: Refactoring Software, Architectures, And Projects In Crisis

Synopsis

"The AntiPatterns authors have clearly been there and done that when it comes to managing software development efforts. I resonated with one insight after another, having witnessed too many wayward projects myself. The experience in this book is palpable." -John Vlissides, IBM Research "This book allows managers, architects, and developers to learn from the painful mistakes of others. The high-level AntiPatterns on software architecture are a particularly valuable contribution to software engineering. Highly recommended!" -Kyle Brown Author of The Design Patterns Smalltalk Companion "AntiPatterns continues the trend started in Design Patterns. The authors have discovered and named common problem situations resulting from poor management or architecture control, mistakes which most experienced practitioners will recognize. Should you find yourself with one of the AntiPatterns, they even provide some clues on how to get yourself out of the situation." -Gerard Meszaros, Chief Architect, Object Systems Group Are you headed into the software development mine field? Follow someone if you can, but if you're on your own-better get the map! AntiPatterns is the map. This book helps you navigate through today's dangerous software development projects. Just look at the statistics: * Nearly one-third of all software projects are cancelled. * Two-thirds of all software projects encounter cost overruns in excess of 200%. * Over 80% of all software projects are deemed failures. While patterns help you to identify and implement procedures, designs, and codes that work, AntiPatterns do the exact opposite; they let you zero-in on the development detonators, architectural tripwires, and personality booby traps that can spell doom for your project. Written by an all-star team of object-oriented systems developers, AntiPatterns identifies 40 of the most common AntiPatterns in the areas of software development, architecture, and project management. The authors then show you how to detect and defuse AntiPatterns as well as supply refactored solutions for each AntiPattern presented.

Book Information

Paperback: 336 pages

Publisher: Wiley; 1 edition (April 3, 1998)

Language: English

ISBN-10: 0471197130

ISBN-13: 978-0471197133

Product Dimensions: 7.7 x 1 x 9.6 inches

Shipping Weight: 1.6 pounds (View shipping rates and policies)

Average Customer Review: 3.5 out of 5 stars   51 customer reviews

Customer Reviews

If patterns are good ideas that can be re-applied to new situations, AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis looks at what goes wrong in software development, time and time again. This entertaining and often enlightening text defines what seasoned developers have long suspected: despite advances in software engineering, most software projects still fail to meet expectations--and about a third are cancelled altogether. The authors of AntiPatterns draw on extensive industry experience, their own and others, to help define what's wrong with software development today. They outline reasons why problem patterns develop (such as sloth, avarice, and greed) and proceed to outline several dozen patterns that can give you headaches or worse. Their deadliest hit list begins with the Blob, where one object does most of the work in a project, and Continuous Obsolescence, where technology changes so quickly that developers can't keep up. Some of the more entertaining antipatterns include the Poltergeist (where do-nothing classes add unnecessary overhead), the Boat Anchor (a white elephant piece of hardware or software bought at great cost) and the Golden Hammer (a single technology that is used for every conceivable programming problem). The authors then proceed to define antipatterns oriented toward management problems with software (including Death by Planning and Project Mismanagement, along with several miniature antipatterns, that help define why so many software projects are late and overbudget). The authors use several big vendors' technologies as examples of today's antipatterns. Luckily, they suggest ways to overcome antipatterns and improve software productivity in "refactored solutions" that can overcome some of these obstacles. However, this is a realistic book, a mix of "Dilbert" and software engineering. A clever antidote to getting too optimistic about software development, AntiPatterns should be required reading for any manager facing a large-scale development project. --Richard Dragan

Patterns are popular in software development and used to identify different types of procedures, designs, or codes that work. AntiPatterns are the exact opposite. They target common mistakes, errors, and people issues that can cause a software project to fail. Despite its negative sounding name, the positive benefits of AntiPatterns are enormous. This book discusses what AntiPatterns

are and then provides practical guidelines on how to detect AntiPatterns and the refactored solutions that correct them. The authors discuss over 40 different AntiPatterns in the areas of software development, architecture, and project management.

Highly recommended for all software engineers (and Project Managers). Some patterns won't apply, so don't feel like you'll have to read it straight through, but there are a lot of good patterns and solutions in here. I also like that they specifically call out when an Anti-Pattern might be the best solution for a particular problem. It is a little dated so some of their solutions have been replaced by better solutions in modern languages, but you can always take their examples and just port them to the language of your choice.

This book showed me the reality: problems in management and development. And then a solution appears to every problem. These problems repeat again and again over time. Sadly, we don't stop to review those problems, prefering to omit them than confront. Now, the book give us the description about problems explicity and the related solutions. And, in practice, it's very useful. Commonly observed, the Analysis Paralysis is an anti-pattern for large projects, and cause the major problems in management.The book is the evidence to the existence of very annoying problems, that frecuently are omited and it's consequences are disasterous. I think it's recommended to management and engineering professionals. Possibly applicable to another branches of engineering.

I'd previously bought the Management Anti-patterns book, which is excellent. There were a couple of references to this book that intrigued me, so I followed up and got this one. The loss of one star is that I only found 3-4 of the described anti-patterns useful. The loss of the other star is that the taxonomy is weaker than the management one, so all of the anti-patterns cited in this book are less useful as a result.

A must read, in addition to reading "Design Patterns". A good compilation of reasons and events why many software projects fail in our days. Also recommend reading "How to Fail At Software Development" to compliment this. There are also a few "Design Patterns" books, including the GoF's, and the blonde skinny girl cover "Head-First" series. Looking forward to discussing the anti-patterns with co-workers to disseminiate the info. I assume every software dev. type person read "Code Complete", right?

Love it!

Over fifteen years old and still an incredibly valuable resource. (Because too few programmers have read it...)

This book starts slowly. I put it down more than once before I got as far as the catalog of anti-patterns. Once there, I found some useful information on what goes wrong with software projects and how to address the problem areas.I would characterize this book as "experiences with software engineering" in general. There is a fair amount on architectural principles, design, and even management topics that were worth the read.

AntiPatterns brings to light a previously unaddressed problem with the way software is currently developed. I have found this book very useful in creating a map moving forward to help myself and my team write better software.

AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis Software Engineering: The Current Practice (Chapman & Hall/CRC Innovations in Software Engineering and Software Development Series) Refactoring to Patterns Refactoring: Improving the Design of Existing Code The Software Requirements Memory Jogger: A Pocket Guide to Help Software And Business Teams Develop And Manage Requirements (Memory Jogger) Software Agreements Line by Line, 2nd ed.: A Detailed Look at Software Agreements and How to Draft Them to Meet Your Needs Head First Software Development: A Learner's Companion to Software Development Agile Project Management: Agile Revolution, Beyond Software Limits: A Practical Guide to Implementing Agile Outside Software Development (Agile Business Leadership, Book 4) Don't Buy Software For Your Small Business Until You Read This Book: A guide to choosing the right software for your SME & achieving a rapid return on your investment IEC 62304 Ed. 1.0 b:2006, Medical device software - Software life cycle processes Agile Software Development with Scrum (Series in Agile Software Development) Corrections and Collections: Architectures for Art and Crime Radio Receiver Technology: Principles, Architectures and Applications VLSI Test Principles and Architectures: Design for Testability (The Morgan Kaufmann Series in Systems on Silicon) Wireless and Mobile Network Architectures Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation Stud: Architectures of Masculinity Prose Architectures High-Performance Computing on

the IntelÃ Â® Xeon PhiÃ¢ Â¢: How to Fully Exploit MIC Architectures Architectures of Humanitarian Space