Software Engineering: A Practitioner's Approach

Roger S. Pressman

Software Engineering: A Practitioner's Approach

Roger S. Pressman

Software Engineering: A Practitioner's Approach Roger S. Pressman
"Software Engineering: A Beginner's Guide" is designed so that it may be used as a supplementary text for any introductory programming course or as the primary text for an introductory software engineering course. Because the book is programming language independent, it can be used effectively regardless of the programming language that is used for instruction. The book can also be used for any course that requires software project work in which the instructor demands application of software engineering principles.

Software Engineering: A Practitioner's Approach Details

Date   : Published January 23rd 2014 by McGraw-Hill Education (first published June 28th 1987)
ISBN   : 9780078022128
Author : Roger S. Pressman
Format : Hardcover 976 pages
Genre   : Science, Computer Science, Reference, Nonfiction, Programming, Technology, Technical

⬇ **Download** Software Engineering: A Practitioner's Approach ...pdf

📄 **Read Online** Software Engineering: A Practitioner's Approach ...pdf

Download and Read Free Online Software Engineering: A Practitioner's Approach Roger S. Pressman

From Reader Review Software Engineering: A Practitioner's Approach for online ebook

Irfin says

I used this book in the first year in my software engineering master degree. Nowadays I still used this book as a reference in my software development.

Any serious software developer who called themselves a software engineer, must read this book.

Kasper Christiansen says

What is the point of writing a book only to fill the pages with references to other peoples work? If Mr. Pressman was able to sum up the referenced articles to a satisfying degree, this might have been worth the read, but all he does is briefly describe the concepts and then pointing to other peoples discussions about it. This book ends up failing in my opinion because it touches a lot of subjects, without ever delivering enough substance (by itself).

If you are looking for a book to point you in the direction of subject relevant articles/papers, this might be the book for you, sadly I was expecting to find some substance here.

Will Vasconcelos says

A great book for learning how to organize software engineering from requirements gathering to implementation and maintenance. However, this book is not about algorithms or programming techniques and will focus solely on documentation primarily through UML.

Nathaniel Mathews says

If you parse this book properly, there's some helpful ideas in it. But the overview it provides would be typical of a book one fourth the length of Pressman's. The bulk of the text is due to buzzwords and repetition rather than depth. Software Engineering is an important topic, but there are better books to learn it from than this one.

Ghamdan says

it is anice book

Ravi Ugale says

i think this book is very funtastic

Alexi Parizeau says

It's not often that I find a textbook which I love this much; it's such a wonderfully lucid journey through software engineering. The writing style is contemporary and compelling, and the content is structured in a naturally linear order, so if software engineering is your thing you might find it a real good page-turner from cover to cover.

Omprakash says

I THINK VERY USEFUL

John says

This is one of the best books on software engineering!

David says

This was a pretty comprehensive (if over-wordy) book back when I was in university. It tries to highlight the current best practices of software development methodology. Unfortunately, it probably doesn't highlight the really up-to-date stuff. Even back then it was more focused on structured design rather than object-oriented best practices, such as doing white box testing and data flow diagrams. There are some gems in there, like the discussions on metrics and formal methods, but the writing is somewhat verbose, making it difficult to sift through.

Mike McDougal says

The organization of this textbook is terrible. Reading and incorporating the facts presented in this book is similar to having a conversation with a drunk person trying to give advice. This is probably the worst textbook I've ever been assigned to read. The book does contain good information, but the presentation is completely dreadful.

Beyond the dubious format, the anecdotal examples presented in the form of a play are worthless. They seem to only serve as filler in this nearly 1000 page long book. One can only assume these mini-plays were added in an attempt to create a market for new editions to be sold. We can't have students buying too many used

textbooks, now can we?

I am eagerly anticipating the completion of this course. The textbook has made this one of the worst courses in my academic experience.

Milad Khajavi says

?? ??? ?? ??? ???? ?? ?? ?????? ???? ???????? ????? ? ?? ??? ?? ?? ??? ?? ?? ????? ??????? ???.
?? ???? ??????? ????? ????? ????? ?? ?????? ?????? ??? ?? ??? ?????.
?? ????? ??? ???? ??? ?????? ???? ???? ????? ? ?????? ????????? ?????????? ???? ???? ??? ?? ??? ????? ?? ???? ????:
?- ???? ???? ???? ????? ???
?- ??? ???? ?? ??? ????? ?????? ?????? ???? ??????? ???? ????? ???? ??? ???????? ?????? ??????? ?? ??? ?????. ??????? ???? ??? ???? ????????? ?? ????? ?????? ??? ??? ?? ?????? ?????? ????????? ?? ?? ??? ???? ???????.
?- ???? ?? ?????? ??????? ????? ??? ? ???? ??????? ?????
?- ???? ?? ?????? ????? ?????? ??? ???? ?? ???? ?? ????? ????? ?? ??? ???.
?- ???????? ??????? ??? ???? ????? ?????? ????? ???? ?? ????? ?? ?? ?? ?? ???? ???? ???. ???? ??? ?????.
?- ??? ???? ???? ??????? ????.
?- ?? ??????? ?????? ????? ??????? ??? ???.
?- ???? ????? ???? ??????? ???? ?? ?????? ??????? ?? ?? ?????? ? ???? ????? ????? ??? ?? ??????? ??? ?? ?? ????????.

????? ??????? says

Best kick start reference for software development I stumbled upon. You don't have to read it all to get what you want.

Adam Turoff says

A poor textbook for a murky field of study.

Sadly, software engineering is a big grab bag of topics necessarily focuses on how high ceremony, high discipline projects work. Real projects almost never see the kinds of specialization and formalism described in a software engineering class, except for big ticket, high cost, cannot fail, bureaucratic projects. Unfortunately, those projects have very little in common with each other, and even less in common with everyday projects, especially those using any of a number of agile methods.

Yet the text paints everything with a broad brush, giving the impression that every project should have a staff of 20 dotting i's and crossing t's, while simultaneously giving the impression that you can choose any of a, b, c, ..., q, r, s, ... w to structure your project, oh wait, you need to do some of (a, b, c) and some of (d, e, f), oh, wait, those steps are optional too, maybe.

The text reads like an accretion of paragraphs written over several revisions, never edited for readability,

consistency, currency, applicability or accuracy, just so long as it has "plenty of examples" and drops enough references to notable names, papers and approaches to appear comprehensive. Without ever vetting that the tools, techniques and examples references are in any way actually relevant or in use.

Hardly a practitioner's approach to software engineering.

Lyle says

This was the college textbook for one of my Software Engineering courses. It's pretty thorough, and is a good (if dated) introduction to the topic of Software Engineering, but not exactly a page turner.