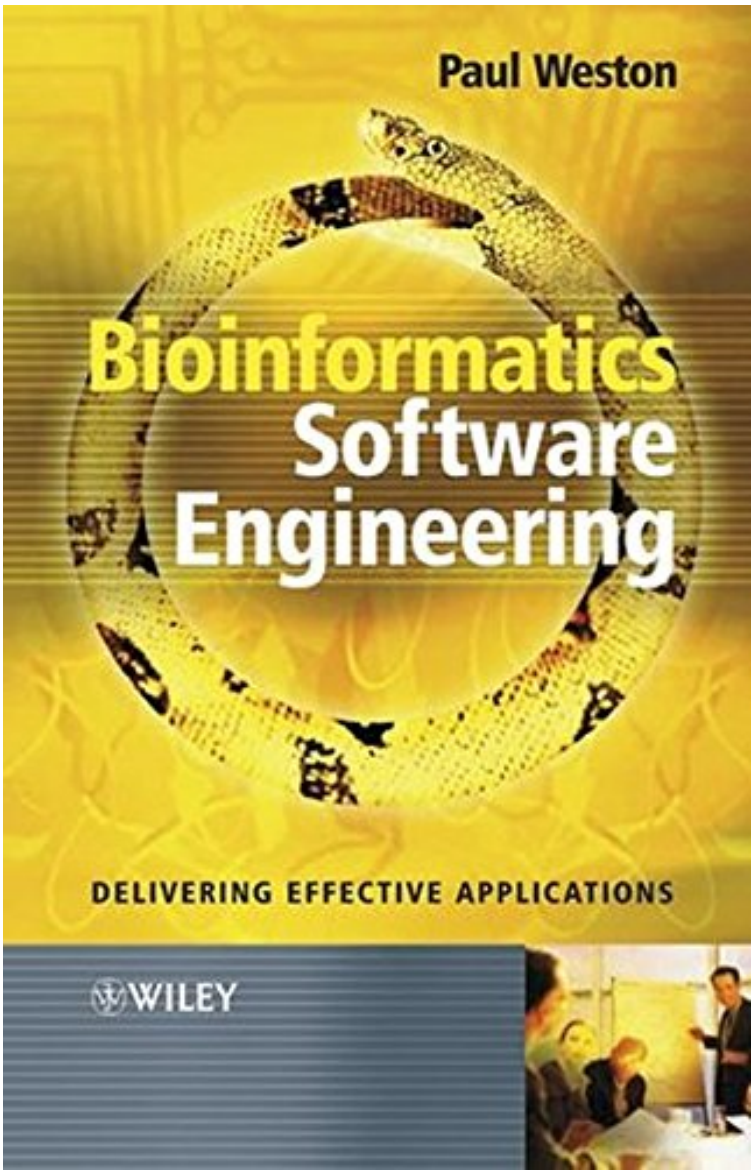
Bioinformatics Software Engineering: Delivering Effective Applications

Paul Weston



#5695123 in Books 2004-11-30Original language:EnglishPDF # 1 9.60 x .41 x 6.70l, .70 #File Name: 0470857722140 pages | File size: 39.Mb

Paul Weston : Bioinformatics Software Engineering: Delivering Effective Applications before purchasing it in order to gage whether or not it would be worth my time, and all praised Bioinformatics Software Engineering: Delivering Effective Applications:

5 of 5 people found the following review helpful. Okay...for the naive biologistBy Calvin ProfessorI sat down and read

through it at a recent bioinformatics conference in Detroit where Wiley had an exhibit. It's applicable for people who just learned how to code and have not had any formal software development training. In my experience, this type of scenario happens a lot: an amateur buys something such as Visual Basic and suddenly he or she is writing code. Someone in the office hears about it and things begin to explode. Suddenly, there is a ton of poorly tested, poorly engineered code, but the amateur is seen as an incredible genious because they can fix the mistakes they made - mistakes that probably should have never happened. So if you have someone in your office who got their hands on a compiler for the first time, you might want to read this to find out that there is more to programming than just writing code. I thought the text was well written for both managers and novice coders.However, if you're an experienced developer (ie you have a four year degree in CS or are familiar with concepts of software testing, requirements gathering etc)you won't find any great insights in this one that makes it worth the price. It's basic software engineering information for software developers getting started.As far as my comment on price, I'll add one more thing. If software development is completely new to you, this is a good start. After all, if it saves you from making some of those horrible mistakes I alluded to or opens your eyes so that it encourages you to take a formal course in software engineering and testing, then this is even more justification to encourage your boss to buy this text for you.Again, this is a bit of a mixed bag for a review, and I'm a bit prejudiced about appropriate text books...again, maybe the easy read will encourage some beginners not to take the "cowboy" approach to software development. Get this book as a start...and then get some books with much more depth in critical subject areas such as those by Shari Pfleeger, Len Bass and Paul Jorgensen.Part of me says give it two stars, part of me says give it five stars. It really depends on your background. Just don't expect this book to make you an expert in bioinformatics...it's basic software engineering with bioinformatics as the backround subject.

Bioinformatics Software Engineering: Delivering Effective Applications will be useful to anyone who wants to understand how successful software can be developed in a rapidly changing environment. A handbook, not a textbook, it is not tied to any particular operating system, platform, language, or methodology. Instead it focuses on principles and practices that have been proven in the real world. It is pragmatic, emphasizing the importance of what the author calls Adaptive Programming - doing what works in your situation, and it is concise, covering the whole software development lifecycle in one slim volume. At each stage, it describes common pitfalls, explains how these can be avoided, and suggests simple techniques which make it easier to deliver better solutions. "Well thought-out ... addresses many of the key issues facing developers of bioinformatics software." (Simon Dear, Director, UK Technology and Development, Bioinformatics Engineering and Integration, Genetics Research, GlaxoSmithKline) Here are some examples from the book itself. On software development: Writing software properly involves talking to people often lots of people and plenty of non-coding work on your part. It requires the ability to dream up new solutions to problems so complicated that they are hard to describe. From description to specification: Look for verbs action words, such as does, is and views. Identify nouns naming words, like user, home and sequence. List the adjectives describing words, for example quick, simple or precise. The verbs are the functions that must be provided by your application. The nouns define the parameters to those functions, and the adjectives specify the constraint conditions under which your program must operate. On how to start writing software: Handle errors. Take in data. Show output. Get going! On testing: It may not be physically possible to test every potential combination of situations that could occur as users interact with a program. But one thing that can be done is to test an application at the agreed extremes of its capability: the maximum number of simultaneous users it has to support, the minimum system configuration it must run on, the lowest communication speed it must cope with, and the most complex operations it must perform. If your program can cope with conditions at the edge of its performance envelope, it is less likely to encounter difficulties in dealing with less challenging situations. On showing early versions of software to users: It can be hard explaining the software development process to people who are unfamiliar with it. Code that to you is nearly finished is simply not working to them, and seeing their dream in bits on the workbench can be disappointing to customers, especially when they were expecting to be able to take it for a test drive. On bugs: If your users find a genuinely reproducible bug in production code, apologize, fix it fast, and then fix the system that allowed it through. And tell your customers what you are doing, and why, so they will be confident that it will not happen again. Everybody makes mistakes. Dont make the same ones twice. And one last thought on successful software development: "You have to be a detective, following up clues and examining evidence to discover what has gone wrong and why. And you have to be a politician, understanding what people want, both in public and in private, and how this is likely to affect what you are trying to do. This book cannot teach you how to do all of that, but it can help."

From the Back CoverBioinformatics Software Engineering: Delivering Effective Applications will be useful to anyone who wants to understand how successful software can be developed in a rapidly changing environment, especially newly qualified bioinformaticians. It is not tied to any particular operating system, platform, language, or methodology. Instead it focuses on principles and practices that have been proven in the real world. It is pragmatic, emphasizing the importance of what the author calls Adaptive Programming - doing what works in your situation. It is

concise, covering the whole software development lifecycle in one slim volume. And, unusually for a computer book, it is eminently readable. "Well thought-out ... addresses many of the key issues facing developers of bioinformatics software." Simon Dear, Director, UK Technology and Development, GlaxoSmithKlineAbout the AuthorPaul Weston has nearly two decades experience in application development, gained in environments as diverse as entrepreneurial start-ups and monolithic bureaucracies. From MVS to XP, from COBOL to Java, and from Structured Programming to Struts - he has wrestled with them all.He began developing bioinformatics applications in the mid-1990s and has particular expertise in sequence assembly and sequence data management. He is now a director of Woodcock Stewart, a consultancy specializing in bioinformatics software development and developer training.