

**developer.\***

The Independent Magazine for Software Professionals

## Book Review

### **Agile Documentation**

#### **A Pattern Guide to Producing Lightweight Documents for Software Projects**

By Andreas Rüping

John Wiley & Sons, 2003, 244 pages (ISBN 0470856173)

Review By Donna L. Davis

#### **Are software developers simply bad writers?**

The prevalent belief that software developers cannot write is a sweeping generalization that is both offensive and untrue. That's not to say, however, that software developers haven't produced some shoddy work and slapped the label documentation on it.

#### **Why is documentation so bad?**

Why does the industry have such a horrendous track record where documentation is concerned? The problem is a vicious cycle that goes something like this: (1) Documentation is a long-winded, poorly organized, outdated, erroneous pile of worthless, tree-killing drivel that no one actually takes the time to read. (2) Believing that any effort expended in writing documentation that will not be read is an exercise in futility, the unfortunate soul assigned to write it expends as little effort as possible, opting to impress management (who won't read it anyway) with quantity (since quality might require more work), resulting in worthless, tree-killing drivel. And the beat goes on.

This pattern is compounded by the reality that in many small shops, the software developer is a one-man band, pumping out a rhythmic stream of code while simultaneously wheezing in his testing juice harp, and finally, banging together an after-thought of documentation, like symbols clanging between his knees. Individually, he could play each role with elegance and grace. Ask him to do it all, and documentation will almost certainly rank below working code as a concern. In fact, it barely generates a blip on the "I care" meter.

In larger, well-funded organizations, the response is to employ professional technical writers, who produce slick-looking documents. However (and I think I am qualified to say this, having a master's degree in technical communication as well as being a software developer), the bigger the gap between the knowledge source and the writer, the higher the probability of producing useless documentation. (This is less true for end-user manuals that benefit from the fresh, unspoiled perspective that a technical writer can bring.)

Thus, in the software development food chain documentation is the murky green fungus that grows on inside of the development tank, sucking the oxygen from unwitting developers who are forced to write it or, worse yet, read it.

Rüping puts it a tad more elegantly, quoting Jerry Weinberg from *The Psychology of Computer Programming*, "Documentation is the castor oil of programming. Managers think it is good for programmers and programmers hate it!"

### **The Agile Answer: Minimum Necessary**

In *Agile Documentation*, Rüping gets to the heart of the documentation dilemma, offering a two-word solution (with a few extra words of helpful explanation): minimum necessary.

Useful documentation has its place, but it should be succinct, worded simply, and presented well. That's where Rüping's book comes in, laying the foundation with elegant simplicity. The book itself is a testament to the layout the author endorses, with strategic use of white space, helpful margin sub-headings and a few useful examples drawn from credible project experience. Issues are laid out in readable fashion, broken down into sections for Problem, Forces, Solution, and Discussion, making the book suitable for flip-through scanning.

Agile Documentation includes topics such as:

- Target audience
- Focus
- Requirements
- Knowledge reuse
- Document categories: management, specification, design, migration, test, usage, and operations

- Long-term Relevance
- Design Rationale
- Realistic Examples
- Structuring documents
- Accessibility
- Diagrams and tables
- Document history
- Layout and typography
- Organization
- Document Templates
- Tools

## It's All About Quality

For many software developers, Rüpings' guidelines will seem surprisingly familiar, reinforcing the notion that for the most part, we recognize quality documentation when we see it. We simply need to pay attention to the details and apply the discipline necessary to produce the same high quality we demand from our software products. Rüpings again quotes Gerald Weinberg, saying, "The value of documentation is only to be realized if the documentation is well done. If it is poorly done, it will be worse than no documentation at all."

Finally, Rüpings includes a healthy-sized chapter on Management and Quality Assurance, highlighting two critical steps that are often dismissed in the glee of typing the final word on the page: review and marketing.

Stakeholders must be given the opportunity to review and provide helpful feedback—an activity that is more fruitful when a "positive review culture" has been fostered, with both author and reviewer understanding and being open to their respective roles.

Still feel like that technical masterpiece is going to languish in unread obscurity like the half-finished novel in your bureau drawer? Rüpings' got it covered in his discussion of the "information marketplace." Again, even though "give it to the target audience" or "post it on a bulletin board" seem like "Well, duh" suggestions, Rüpings knows (like we do, if we stop and think about it), these obvious measures are often not done.

## Is Documentation All Bad?

Documentation isn't bad. But bad documentation is terrible. Follow Rüping's advice in Agile Documentation and cut the fat from your documentation diet.

###

Donna Davis is a database administrator, project manager, programmer, supervisor, and professional author living and working in North Carolina, US. She can be reached through the editorial staff of **developer.\***.