<<敏捷开发艺术>>

图书基本信息

书名：<<敏捷开发艺术>>

13位ISBN编号：9787564112417

10位ISBN编号：7564112417

出版时间：2008-8

出版时间：东南大学出版社

作者：（美）肖尔（Shore,J.）　（美）活登（Warden,S.）　著

页数：409

前言

We want to help you master the art of agile development Agile development, like any approach to team-based software development, is a fundamentally human art, one subject to the vagaries of individuals and their interactions. To master agile development, you must learn to evaluate myriad possibilities, moment to moment, and intuitively pick the best course of action. How can you possibly learn such a difficult skill? Practice! First and foremost, this book is a detailed description of one way to practice agile development: Extreme Programming (XP). It ' s a practical guide that, if followed mindfully, will allow you to successfully bring agile development in the form of XP to your team—or will help you decide that it ' s not a good choice in your situation. Our second purpose is to help you master the art of agile development. Mastering agility means going beyond our cookbook of practices. Agile development is too context-sensitive for one approach to be entirely appropriate, and too nuanced for any book to teach you how to master it. Mastery comes from within: from experience and from an intuitive understanding of ripples caused by the pebble of a choice. We can ' t teach you how your choices will ripple throughout your organization. We don ' t try. You must provide the nuance and understanding. This is the only way to master the art. Follow the practices. Watch what happens. Think about why they worked... or didn ' t work. Then do them again. What was the same? What was different? Why? Then do it again. And again. .. At first, you may struggle to understand how to do each practice. They may look easy on paper, but putting some practices into action may be difficult. Keep practicing until they ' re easy. As XP gets easier, you will discover that some of our rules don ' t work for you. In the beginning, you won ' t be able to tell if the problem is in our rules or in the way you ' re following them. Keep practicing until you ' re certain. When you are, break the rules. Modify our guidance to work better for your specific situation. Parts I and II of this book contain our approach to XP. Part I helps you get started with Extreme Programming; Part II provides detailed guidance for each of XP ' s practices. Parts I and II should keep you occupied for many months. When you ' re ready to break the rules, turn to Part III. A word of warning: there is nothing in Part III that will help you practice XP. Instead, it ' s full of ideas that will help you understand XP and agile development more deeply. One day you ' ll discover that rules no longer hold any interest for you. After all, XP and agile development aren ' t about following rules. " It ' s about simplicity and feedback, communication and trust, " you ' ll think. " It ' s about delivering value—and having the courage to do the right thing at the right time. " You ' ll evaluate myriad possibilities, moment to moment, and intuitively pick the best course of action. When you do, pass this book on to someone else, dog-eared and ragged though it may be, so that they too can master the art of agile development.

<<敏捷开发艺术>>

内容概要

本书是讲解如何利用敏捷开发方法构建高价值软件的实用指南，描述了什么是敏捷开发，以及它能帮助软件项目获得成功的原因。
该书还将开发者、项目经理、测试者和客户所需信息整合在一起，以便直接运用。
《敏捷开发艺术》展现了敏捷过程的完整视图，基于作者多年的极限编程（XP）经验，直截了当地提出关于计划、开发、交付和管理等多方面实施的建议。
它为开发者和测试者提供实用的技术练习，同样也为非技术背景读者提供了充分的信息。
作者还介绍了如何处理敏捷开发中的棘手问题：建立团队成员之间的协作和信任关系。
《敏捷开发艺术》针对下列问题给出明确的答案：　如何采用敏捷开发？
我们是否真的需要结对编程？
应该基于何种度量（metrics）报告？
如何让我们的客户共同参与项目？
我们应该撰写多少文档？
何时设计架构？
作为非开发者，该如何与敏捷团队合作？
我的产品路线图在哪里？
QA如何适应敏捷开发？
无论你是敏捷团队的一员，还是刚刚对敏捷开发产生兴趣，这本书都具备了你需要的所有实用技巧。
它向你说明引入XP的过程，详细描述其中每一项实践，并且讨论了如何修改xP和创建自己的敏捷方法等相关原则。
该书将随着你的经验提升而不断深入，首先教你规则，然后告诉你如何突破它们，当你掌握了敏捷开发艺术之时，最终便可以摈弃一切规则。

作者简介

作者：(美国)肖尔 (James Shore) (美国)活登 (Shane Warden)

<<敏捷开发艺术>>

书籍目录

章节摘录

Two customers for every three programmers seems like a lot, doesn't it? Initially I started with a much smaller ratio, but I often observed customers struggling to keep up with the programmers. Eventually I arrived at the two-to-three ratio after trying different ratios on several successful teams. I also asked other XP coaches about their experiences. The consensus was that the two-to-three ratio was about right.Most of those projects involved complex problem domains, so if your software is fairly straightforward, you may be able to have fewer customers. Keep in m ind that customers have a lot of work to do. They need to figure out what provides the most value, set the appropriate priorities for the work, identify all the details that programmers will ask about, and fit in time for customer reviews and testing. They need to do all this while staying one step ahead of the programmers, who are right behind them, crunching through stories like freight trains. It's a big iob. Don't underestimate it. The product manager has only one job on an XP project, but it's a doozy. That job is to maintain and promote the product vision. In practice, this means documenting the vision, sharing it with stakeholders, incorporating feedback, generating features and stories, setting priorities for release planning, providing direction for the team's on-site customers, reviewing work in progress, leading iteration demos, involving real customers, and dealing with organizational politics.The best product managers have deep understandings of their markets, whether the market is one organization (as with custom software) or many (as with commercial software). Good product managers have an intuitive understanding of what the software will provide and why it's the most important thingtheir project teams can do with their time. A great product manager also has a rare combination of skills. In addition to vision, she must have the authority to make difficult t ade-off decisions about what goes into the product and what stays out. She must have the political savvy to align diverse stakeholder interests, consolidate them into the product vision, and effectively say "no" to wishes that can't be accommodated.Product managers of this caliber often have a lot of demands on their time. You may have trouble getting enough attention. Persevere. Theirs is one of the most crucial roles on the team. Enlist the help of your project manager and remind people that software development is very expensive. If the software isn't valuable enough to warrant the time of a good product manager——a product manager who could mean the difference between success and failure——perhaps it isn't worth developing in the first place.Make sure your product manager is committed to the project full-time. Once a team is running smoothly, the product manager might start cutting back on his participation. Although domain experts and other on-site customers can fill in for the product manager for a time, the project is likely to start drifting off-course unless the product manager participates in every iteration. [Rooney] experienced that problem, with regrettable results:We weren't sure what our priorities were. We weren't exactly sure what to work on next. We pulled stories from the overall list, but there was precious little from the Customer [product anager] in terms of what we should be working on. This went on for a few months.Then, we found out that the Gold Owner [executive sponsor] was pissed——really pissed. We hadn't been working on what this person thought we should.In a predictable environm nt, and by delegating to a solid set of on-site customers, a product manager might be able to spend most of his time on other things, but he should still participate in every retrospective, every iteration demo, and most release planning sessions.Some companies have a committee play the role of product manager, but I advise against this approach.The team needs a consistnt vision to follow, and I've found that committees have trouble creating consistent, compelling visi ns. When I've seen committees succeed, it's been because one committee member acted as de facto product manager. I recommend that you explicitly find a product manager.Her role may be nothing more than consolidating the ideas of the committee into a single vision, and that's likely to keep her hands full. Be sure to choose a product manager with plenty of political acumen in this case.Most software operates in a particular industry, such as finance, that has its own specialized rules for doing business. To succeed in that industry, the software must implement those rules faithfully and exactly. These rules are domain rules, and knowledge of these rules is domain knowledge.Most programmers have gaps in their domain knowledge, even if they've worked in an industry for years. In many cases, the industry itself doesn't clearly define all its rules. The basics may be clear, but there are nitpicky details where domain rules are implicit or even contradictory.The team's domain experts are

responsible for figuring out these details and having the answers at their fingertips. Domain experts, also known as subject matter experts, are experts in their field. Examples include financial analysts and PhD chemists.Domain experts spend most of their time with the team, figuring out the details of upcoming stories and standing ready to answer questions when programmers ask. For complex rules, they create customer tests (often with the help of testers) to help convey nuances. The user interface is the public face of the product. For many users, the UI is the product. They judge the product's quality solely on their perception of the UI.Interaction designers help define the product UI. Their job focuses on understanding users, their needs, and how they will interact with the product. They perform such tasks as interviewing users, creating user personas, reviewing paper prototypes with users, and observing usage of actual software.NOTE Don't confuse graphic design with interaction design. Graphic designers convey ideas and moods via images and layout. Interaction designers focus on the types of people using the product, their needs, and how the product can most seamlessly meet those needs. You may nothave a professional interaction designer on staff. Some companies fill this role with a graphic designer, the product manager, or a programmer. Interaction designers divide their time between working with the team and working with users. They contribute to release planning by advising the team on user needs and priorities. During each iteration, they help the team create mock-ups of UI elements for that iteration's stories. As each story approaches completion, they review the look and feel of the UI and confirm that it works as expected. The fast, iterative, feedback-oriented nature of XP development leads to a different environment than interaction designers may be used to. Rather than spending time researching users and defining behaviors before development begins, interaction designers must iteratively refine their models concurrently with iterative refinement of the program itself. Although interaction design is different in XP than in other methods, it is not necessarily diminished. XP produces working software every week, which provides a rich grist for the interaction designer's mill. Designers have the opportunity to take real software to users, observe their usage patterns, and use that feedback to effect changes as soon as one week later. On nonagile teams, business analysts typically act as liaisons between the customers and developers, by clarifying and refining customer needs into a functional requirements specification. On an XP team, business analysts augment a team that already contains a product manager and domain experts. The analyst continues to clarify and refine customer needs, but the analyst does so in support of the other on-site customers, not as a replacement for them. Analysts help customers think of details they might otherwise forget and help programmers express technical trade-offs in business terms.A great product vision requires solid execution. The bulk of the XP team consists of software developers in a variety of specialties. Each of these developers contributes directly to creating working code. To emphasize this, XP calls all developers programmers.

媒体关注与评论

“ 我会将此书送给我访问过的每一个团队。
”—— Brian Marick， Exampler Consulting ...

编辑推荐

《敏捷开发艺术(影印版)》由东南大学出版社出版。

版权说明