INTRODUCTION

Provides background information to help you get the most out of the book

from

Flexible Product Development

by

Preston G. Smith

Introduction

This book's initial chapter, "Understanding Flexibility," opens the subject, introduces many key concepts, and sensitizes you to important points as you read the rest of the book. I therefore suggest that you read it first.

Chapters Two through Nine describe the tools, techniques, approaches, and strategies of flexible development. Think of them as a kit of tools. Just like the tools you might use to fix your car or repair the plumbing, you will not use all of the tools in your kit on every job, and certain ones are inappropriate under certain circumstances. I try to emphasize the limitations and inappropriate uses, but my advice here is limited, because I cannot envision all the applications that you may face.

These eight chapters are in largely arbitrary order. I have tried to keep them independent, and you could read them in any order and skip ones that do not seem to apply to your operations. Nonetheless, there is some order to the chapter arrangement. The chapter on customers comes first, because good product development always starts with the customer. Then I cover the core techniques of product architecture, experimentation, and set-based design to expose you to the meat of the book early.

Next I present a chapter that is also central to high-performance product development: development teams. You probably have already read plenty on teams, so I concentrate on what supports flexibility in teams. Because co-located teams are so critical to flexibility, these—and their opposites, globally dispersed teams—receive special attention. A chapter on decision making

follows the one on teams, because most of the decisions involved are made within the team and depend on the strong communication channels that result from the practices in the teams chapter.

The next two chapters—on project management and development processes—appear at the end of the book for two reasons. First, flexibility de-emphasizes structured processes and mainstream project management techniques, such as work breakdown structures, so these topics, in a certain sense, come last. Second, these are complex subjects that require an understanding of the nature of flexibility, which will come from the preceding chapters, so the earlier chapters will be helpful prerequisites.

I follow these eight toolkit chapters with a chapter on implementation. This is a critical chapter with a sobering objective, for it asserts that none of the tools and techniques in the other chapters will be of any use in your business until you implement them. This often requires significant changes in values, so I provide an effective approach for making such transitions.

Finally, this revision provides a chapter on managing anticipated changes. It presumes that when you are starting a project, you can already identify one or two changes that you anticipate, so this chapter provides a technique for managing specific changes that you are aware of initially. This is a supplement, not a substitute for the rest of the book, which creates an environment in which any change can be accommodated without undue disruption.

One important approach does not have a chapter of its own, because it is so central to flexible development that it pervades all of the others: *iteration* .[1] In contrast with traditional methods, which generally plan activities in advance and execute them sequentially (with some overlap for speed), flexible approaches typically make many small loops through the process, obtaining interim feedback from tests, customers, or management before starting the next loop. Because flexible developers seldom know precisely where they are going—due to the uncertainty that is the hallmark of flexibility—it's essential to start with small steps in what seems to be the right

direction and make adjustments from there. The tools of flexibility, covered in Chapters Two through Nine, will help you to iterate more effectively.

While revising this book, I considered updating the examples. I decided not to; although some examples are laughably dated (see Figure 3.1, for instance), they still work well, and the principles they illustrate are timeless.

Sources

This may be the first book on product development flexibility, but I did not invent the material. Rather, I have packaged techniques from diverse sources. Many of these tools have a rich history and extensive usage in other areas. Thus, I provide comprehensive endnotes to indicate the roots or where you can find related material and a bibliography that supports the endnotes.

Terminology

Why have I chosen to call this *flexibility? Agility* would do just as well, but that word is already in use for the same purpose by the software development community. Because this book must be a rebuilding of what the agilists have done and not merely a translation, I avoid *agile* except in reference to software development.

More important, what is the opposite of *flexibility? Rigidity* first comes to mind, but no one would admit to being rigid. Barry Boehm and Richard Turner, in an otherwise excellent book, use *discipline* in the title and *plan-driven* inside the book as opposites of *agile.* Agilists object to both terms on the basis that agilists have plenty of discipline, but it is of an unconventional variety. And they probably plan more than the plan-driven folks, but it is done where it is unapparent. *Structured* is another possibility, but it has some of the same problems. Consequently, I hope to avoid all these booby traps by using *traditional* as the opposite of *flexible.*