Agile软件开发丛书

Configuration Management Principles and Practice
配置管理原理与实践（影印版）

The Agile Software Development Series

Forewords by
Kim Caputo
and Alistair Cockburn

Alistair Cockburn指导、任丛书编辑并亲自作序

[美] Anne Mette Jonassen Hass 著

清华大学出版社

Agile 软件开发丛书

配置管理原理与实践

（影印版）

[美] Anne Mette Jonassen Hass 著

清华大学出版社

·北京·

内 容 简 介

本书是关于配置管理的综合指南，介绍了配置管理的基本原理和实践活动。主要包括在整个开发过程应用配置管理、不同环境和项目中应用配置管理、集成不同配置管理工具等内容。

本书面向程序开发人员和项目管理人员，书中大量的实践经验将有助于读者更好地管理和交付项目。

EISBN: 0-321-11766-2
Configuration Management Principles and Practice, 1e
Anne Mette Jonassen Hass

图书在版编目（CIP）数据

配置管理原理与实践 Configuration Management Principles and Practice /（美）哈斯著. 影印版. —北京：清华大学出版社，2003
（Agile 软件开发丛书）
ISBN 7-302-06645-0

Ⅰ.配… Ⅱ.哈… Ⅲ.软件—配置—管理—英文 Ⅳ.TP31

中国版本图书馆 CIP 数据核字（2003）第 037468 号

出 版 者：清华大学出版社（北京清华大学学研大厦，邮编 100084）
http://www.tup.com.cn
http://www.tup.tsinghua.edu.cn

责任编辑：汤涌涛

印 刷 者：世界知识印刷厂

发 行 者：新华书店总店北京发行所

开　　本：880×1230 1/32　**印张**：13.125

版　　次：2003 年 6 月第 1 版　2003 年 6 月第 1 次印刷

书　　号：ISBN 7-302-06645-0/TP・4973

印　　数：0001～3000

定　　价：28.00 元

Foreword by Kim Caputo

Solving the problems in configuration management can dramatically reduce the cost of rework, not to mention reduce the number of programmer headaches. I was fortunate to work in a company that did very well with configuration management practices on their proprietary systems. However, when they began software development on open systems, it was not so easy. Things that were second nature, that were so internalized that we didn't have to think about them anymore, suddenly became the things we didn't have the foresight to think about on the new systems. We began to have problems again. We had to relearn things that we thought we had learned before, and it was difficult to go back and learn them all over again. The explanations of the concepts, definitions, roles, and responsibilities in this book would have helped us then.

This book will also help those who have never had the appropriate level of discipline in their workplace for configuration management, especially those who have experienced horror stories like these:

- The Lost Software: "I know I wrote it, but I don't know where I put it."
- The Missing Links: "This used to work, but now it points to code that isn't there anymore."
- Stepping on each other's code: Developers doing different fixes in the same code area, overwriting each other.
- You Can't Go Back Again: New fixes are worse, and there's no "undo" button.
- You Can't Put It Together Again: Dropped a document with no page numbers, or dropped two documents, no titles on pages, which was which?

- Who's on First? What's on Second? Bug reported by customer, but don't know what version they have, don't know what fix to give them.
- "But I Know I Fixed It!"
  - — Customer calls and says, "It's broken."
  - — Programmer makes the fix, but forgets to check-in the change.
  - — Software build is done without the fix (No one audited the baseline).
  - — Exact same software shipped to the customer.
  - — Customer calls and says, "It's still broken."
  - — Programmer says, "But I know I fixed it!"

Configuration management is a cornerstone of software process improvement. (After all, if you can't manage your stuff, how can you tell whether your stuff has improved?) In *CMM Implementation Guide*, I wrote: "In the software industry, many of us have taken steps in software process improvement and made the steps our own, but perhaps many of us have not yet taken the more difficult steps of allowing ourselves to learn from each other and change under cross-cultural influence. It won't happen unless we share our experiences and our techniques. I am sharing my experiences and techniques, not to tell people to do it my way but rather to open the door for us to learn from each other throughout the industry and throughout the world. Perhaps I am not the first to open this door, and I hope that I am not the last. This is an invitation to the dance."

Across the world, nine time zones away, Anne Mette Jonassen Hass has answered the invitation and come through with a wonderful contribution. Here she shares her experiences and techniques for successful configuration management, with several possibilities for solutions that readers can take and make the steps their own. She also includes a wealth of references to reach more information for further learning. I am delighted with this contribution that takes up the call to influence our industry and our world.

—Kim Caputo
Mission Viejo, California

Foreword by Alistair Cockburn

Software configuration management and automated regression testing tools are the two development tools most critical to the success of the agile project. Over the last ten years, the version control and configuration management system was consistently cited to me as the top priority tool to install, both for agility-focused and plan-driven projects. No other tool even came close. (The editor and compiler are so integral that they don't get named.) Teams used to working with a version control and configuration management system refuse to operate without one.

Many teams find that once they have a satisfactory configuration management system in place they can do something more important to their project than merely coordinate their check-ins: They start experimenting with shorter and shorter periods between builds. (This is when the automated regression testing tool becomes important.)

Some teams run fully automated builds every half hour; these also run the suite of unit and system regression tests, post the results on a Web page, and email the owners of any failed code their failing test results! People on these teams report an increase in speed, agility, quality, and personal comfort, knowing they'll learn of unexpected errors within a half-hour of checking in their code.

One company is even experimenting with using such a continuous-build system to synchronize the work between India and the United States. They report that it is helping the two teams stay synchronized with each other across nine time zones.

It is therefore astonishing to see how many teams try to work without a configuration management system. Moreover, it can be frustratingly difficult to find information on the topic.

Anne Mette Hass manages in this book to capture both the heart of the subject and the variations needed in widely varying circumstances—a rare accomplishment. She knows, as you do, that some organizations run with heavy bureaucracy, some with little bureaucracy, some with little formality, some with great formality—and all need configuration management to smooth their collective work. She presents the topic from several angles: the work products, the job roles involved, the organizational issues, the tools, and various levels of formality and bureaucracy. In addition to her insights, Steve Berczuk and Brad Appleton describe, in their appendix, how the terms and practices can be used on the lightest of agile projects.

I have always found this subject daunting, and was pleased to find this text well presented and easy to digest. I could never have written this book; I'm glad that Anne Mette Hass has done it for us.

—Alistair Cockburn
Salt Lake City, Utah

Preface

My Life as a Software Professional

I have two—well, three really—passions in my professional life: test, configuration management, and process improvement.

I started my career as an all-around developer—a little requirements elicitation, a little analysis, a lot of coding and recoding, and some test—more than 20 years ago. During these first professional years, I always loved testing most—making my work run on the computer and enjoying the satisfaction of being told, in a factual and precise way, that something was wrong. This enabled me to carry out the correction and then finally enjoy the privilege of knowing that at least this error was a secret between me and the computer.

My experience grew, and my working teams grew. The problems grew. I wasn't always certain I had produced what I was supposed to and that I had tested everything. And sometimes an error would recur!

I got a job in which I was responsible for system and acceptance test in a company making software for the European Space Agency. For the first time in my then 12-year career, I heard the words *configuration management*. I had no clue as to what it was, but as I spent hours and hours trying to figure it out, discussing it with the person responsible for quality assurance and actually using parts of it in my daily work, I came to understand what a wonderful tool I had.

For the first time, I was able to trace my test cases to the requirements. I was able to tell, at any point, how many requirements I had covered in my test specification and how many were outstanding. I didn't have to encounter the frustration of having made test cases for requirements that weren't going to be implemented. Where I had

forgotten the reason for a turn in the work, I was able to find a previous version of my test specification and see why I had changed it. I loved it!

The last seven years, I've worked as a consultant, spending a good deal of my time on testing assignments of many types in many companies. One of the things I've learned from these assignments is that there is often a difference between what a customer asks for and what he really wants, what he needs (what you want to give him), and what you're able to give him.

Test consultants are often presented with a system to test without the right conditions for performing a professional test. The requirements may be in any state from nonexistent to brilliantly documented, with a pronounced bias toward the former. If requirements are present, they are most often not up to date. This is partly a requirement specification problem and partly a configuration management problem.

Testing requires resources in terms of time and people to perform the test. These resources are often all too scarce. This is a project management problem.

When test consultants plan and perform a test, they need to establish an overview not only of what has to be tested but also how the test is progressing, what errors have been found, and what the state of error correction is. These are configuration management issues.

It's tempting for a consultant to try to deliver what the customer really needs. However, this approach has some limitations and drawbacks. The art is to strike the right balance between what's needed and what's feasible. One of the things to keep in mind as a consultant is to keep up the standards but keep it light. So I try to keep up the configuration management standards as I solve the test assignment—hoping my customer will get an idea of what configuration management is and maybe ask for some assistance in that direction too.

Another part of my time is spent assessing software-producing companies using the BOOTSTRAP maturity model and method. Like the related Capability Maturity Model (CMM), this model includes configuration management. As an assessor in more than 40 assessments, I have time and again seen the blank look in people's eyes when I ask how they perform configuration management. The eyes are rarely less blank if I elaborate and ask about tracing between work products, production of error reports, or other detailed configuration management disciplines.

On the other hand, people are more than willing to talk about problems they've experienced due to lack of control over what is being implemented and tested—and when—and lack of control over what errors have occurred and which ones are being corrected and which are not.

Although configuration management is one of the basic disciplines for sound development (in CMM it is a key process area at level 2), many people go through a

considerable part of their careers without any idea of what it is and how it can ease their everyday tasks, just as I did. So I keep emphasizing its importance and very often recommend it as one of the first disciplines a company should work on when embarking on structured process improvement.

Creation of This Book

In 1999, the Danish organization *Datateknisk Forum*, an association of about 70 software-producing companies, asked me to write a book on configuration management. This was the result of a survey among the members as to what topic they needed a book on. Some of the comments and requirements that came back from the survey were

- How do you incorporate configuration management in the development process?
- How do you handle the fact that different kinds of work products, like documents and code, are treated differently?
- How do you obtain integration between different configuration management tools?
- How do you handle multisite development?
- How do you handle configuration management in relation to object-oriented development—component-based development?

I took on the assignment because in my own experience, configuration management has been of great value, not because I felt I knew much about it theoretically. I know much more now, and I hope I've conveyed some of the understanding, knowledge, and appreciation I've gained during my work on this book. If readers try at least some of the detailed disciplines, I hope they will experience the same enthusiasm about its usefulness that I did.

The book is based on literature as well as experience—and also on attitudes and opinions. It contains a lot of examples, advice, and recommendations that are not to be regarded as The Truth but primarily as the sum of a lot of experience—negative as well as positive.

When I learned that the book was to be published in the Agile Series, I knew little about agile development. But as I studied the values and principles, I found that I had practiced it in parts for years. Agile development is a wonderful idea, and one of the cornerstones of its success is configuration management, so it was a pleasure to be able to contribute to the series with one of my favorite disciplines.

The book may seem a bit heavy to some agilists, but I think it's better to discard some formality and detailed activities deliberately, knowing what one hasn't performed, than to just not perform it out of ignorance. So, agilists and others, read and choose!

Purpose of the Book

This book is not supposed to be a primer in configuration management. It does, however, start with an introduction to fundamental principles, to establish a basic understanding of the concepts used. The main part of the book discusses more advanced issues encountered when configuration management has to be implemented. The overall purpose of the book is twofold:

- To scare those who are engaging in configuration management! The book will give the reader an understanding of the complexity and comprehensiveness of the discipline. *Configuration management is not easy!* If you think it is, you'll be unable to solve its tasks in a professional way.
- To assuage the fear of those who are engaging in configuration management! The book will provide a fundamental understanding of the principles of the discipline, their interrelations and usage. *Configuration management is not difficult!* All you have to do is do it. If you understand it, it's much easier to specify and plan so it fulfills its purpose and becomes manageable.

It's assumed that the reader has some knowledge of other disciplines within software development, such as planning, design, test, and quality assurance.

Thanks

A lot of people have supported the creation of this book. I have no way of mentioning them all. First, I would like to thank the members and the board of *Datateknisk Forum* and my managers, Mr. Jørn Johansen and Mr. Ole Andersen, for believing in the idea and contributing to the contents.

I would also like to thank my colleagues (especially Ms. Elisabeth Broe Christensen and Mr. Robert Olesen), Mr. Lars Bendix of the University of Lund, Sweden, and not least my husband, Finn, for providing many pieces of good advice and good ideas, and for the interest and patience they have shown during my work on the book. My husband's wry way of looking at things is sometimes annoying but always enlightening—thanks, Finn, for being who you are!

The publisher and my editor, Mr. Ross Venables, deserve lots of thanks for their enthusiasm and encouragement, all the way from my first approach through the development of the manuscript to the complete book.

Last but not least, a big thanks goes to my longtime friend Ms. Pernille Lemvig-Fog and my father, Mr. Birger Jonassen, for their great help with the translation of the text into understandable English.

Contents