Software Configuration Management Patterns

Effective Teamwork, Practical Integration

STEPHEN P. BERCZUK
with BRAD APPLETON
Foreword by Kyle Brown

SOFTWARE PATTERNS SERIES

PRAISE FOR *SOFTWARE CONFIGURATION MANAGEMENT PATTERNS*

I think the authors have just created the new "SCM Bible" that will be the new standard and reference manual for SCM-ers and software development professionals for years and years to come!!

—Jeffrey W. Faist, Jeff Faist Consulting Inc.

I'm very glad that people are still writing about SCM. For a while, it seemed that the momentum had shifted away from competence in SCM, and the book writing was following. The organization of this book is quite good, and the content is quite complete. Everything I'd expect from a quality Addison-Wesley piece of work.

—Craig Gardner

I think this is a timely book—right now source control is most definitely a black art, and most teams do it badly, if at all. There are very few books on the subject.

—Dave Thomas, coauthor of The Pragmatic Programmer: From Journeyman to Master

There's a lot of expertise captured here, and I get a sense of really sitting down with someone who understands these issues.

—Linda Rising, author of The Pattern Almanac 2000

I think this is an excellent book. If you're at all serious about software development, you need SCM; this book could make that case and not only convince readers that they need it, but provide enough information that they could immediately apply the patterns on their projects.

—James Noble, coauthor of Small Memory Software: Patterns for Systems with Limited Memory

This book is a good overview of a very important area of current software development projects. I say this as someone who has endured (along with my fellow team members) various struggles with SCM systems in the last several companies where I have worked. There is very little readily available literature in this field, and I believe this book will prove to be very important to anyone working in a medium- to large-sized development team.

—Bernard Farrell, WaveSmith Networks

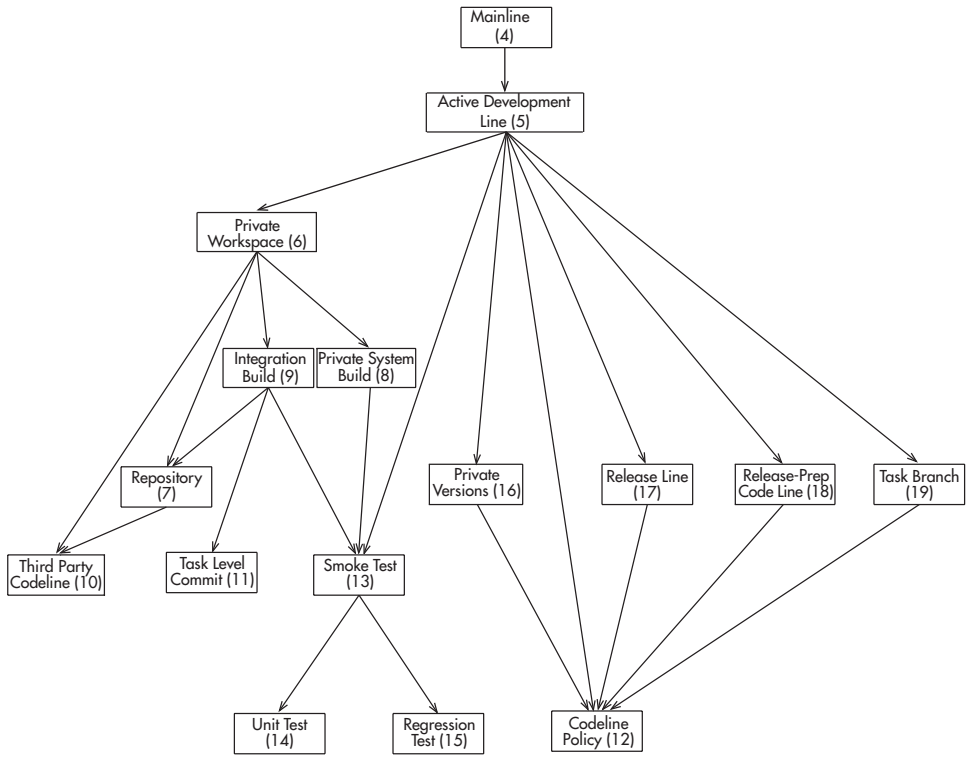

FIGURE 3–1. The SCM pattern language

There are other approaches to development than the *Mainline* approach we discuss here. The *Mainline* approach works well in many circumstances, and other approaches may share many of the same patterns. It is important to remember as you read through these patterns that, although we hope this advice is useful, only you understand your set of circumstances. Apply the patterns with care.

The patterns in the language can be grouped into two sets: codeline-related patterns and workspace-related patterns.

The codeline-related patterns help you organize your source code and other artifacts appropriately, in terms of both structure and time

The patterns relating to codelines are[1]

1. References to patterns show the name of the pattern followed by the chapter number in parentheses.

- *Mainline (4)*
- *Active Development Line (5)*
- *Codeline Policy (12)*
- *Private Versions (16)*
- *Release Line (17)*
- *Release-Prep Code Line (18)*
- *Task Branch (19)*

Figure 3–2 shows these patterns.

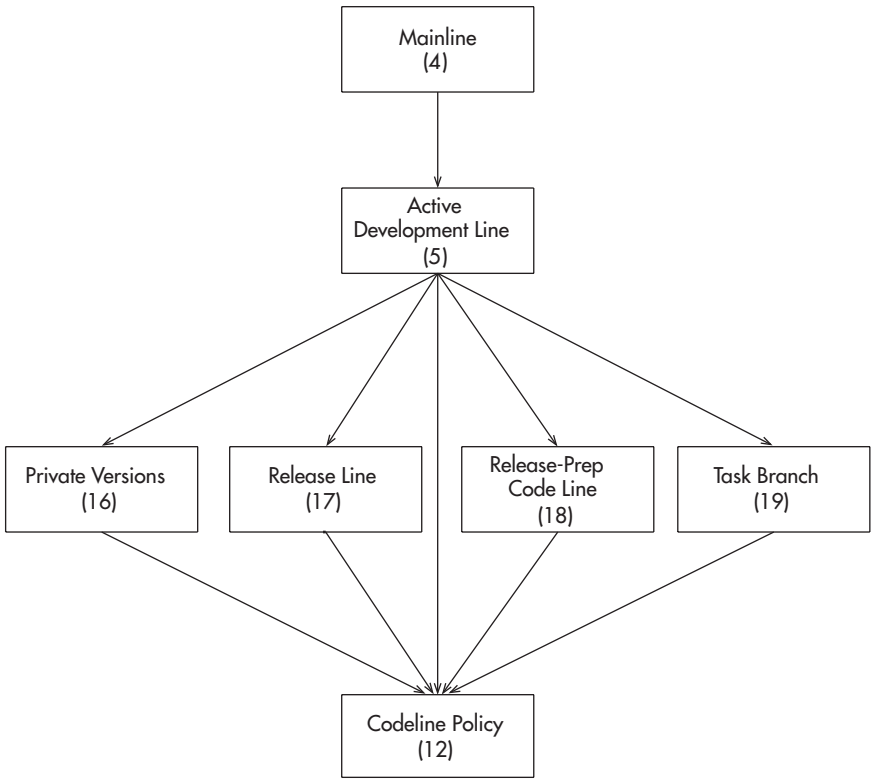

FIGURE 3–2. Codeline-related patterns

Figure 3–3 shows patterns related to workspaces.

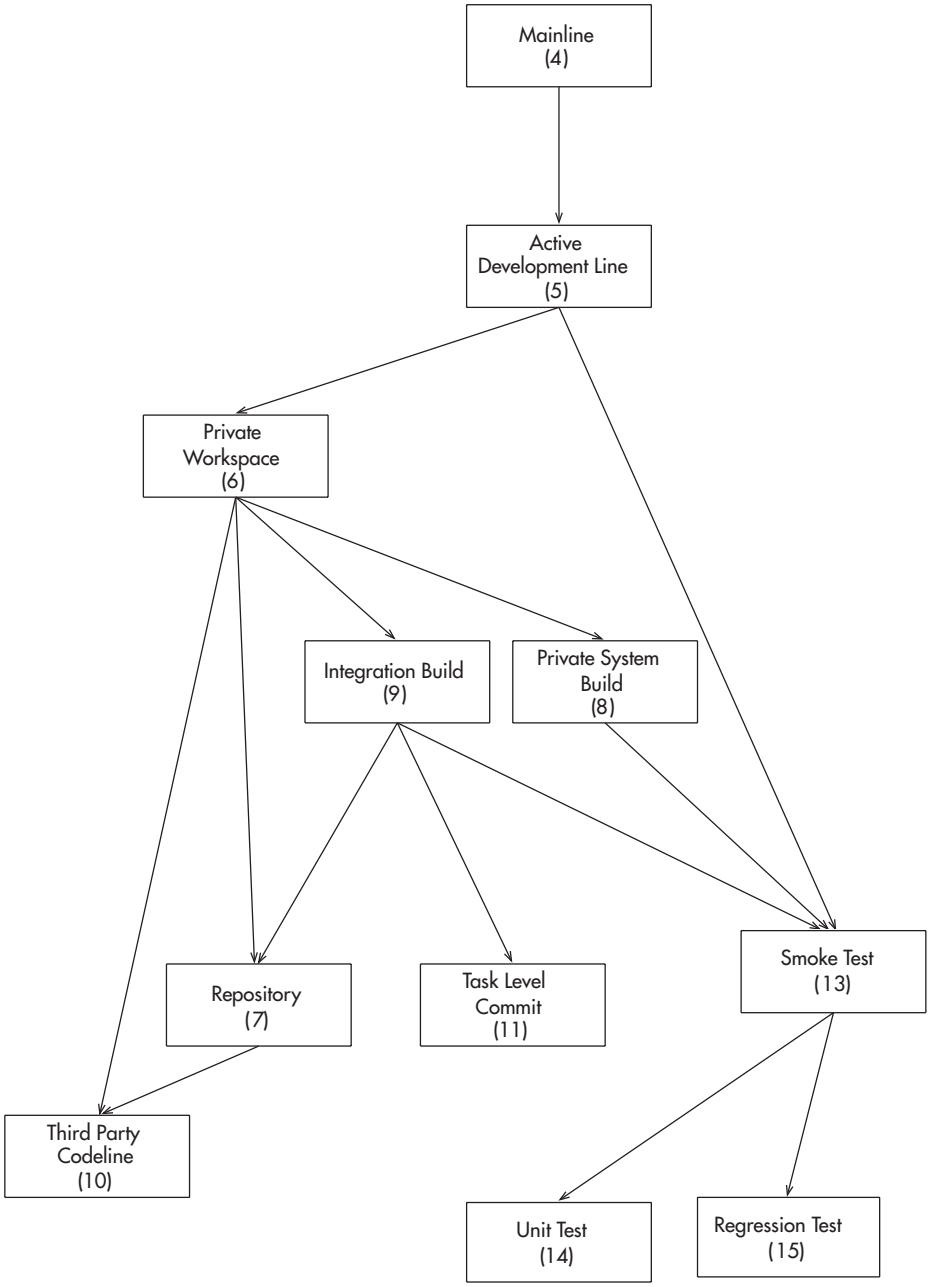

FIGURE 3–3. Workspace-related patterns

The patterns related to workspaces are

- *Private Workspace (6)*
- *Repository (7)*
- *Private System Build (8)*
- *Integration Build (9)*
- *Third Party Codeline (10)*
- *Task Level Commit (11)*
- *Smoke Test (13)*
- *Unit Test (14)*
- *Regression Test (15)*

UNRESOLVED ISSUES

This chapter gives you an overview of pattern languages in general and the pattern language in this book. The rest of the book defines the patterns.

FURTHER READING

- The series of books by Alexander et al. describes the core principles of patterns and pattern languages. *A Pattern Language* (Alexander et al. 1977) gives a pattern language for building towns and buildings. *The Timeless Way of Building* (Alexander 1979) describes the principles behind patterns and pattern languages. *The Oregon Experiment* (Alexander et al. 1975) gives a concrete example of using a pattern language in a real situation. Some of these books are a bit long, and Alexander's prose is a bit tough to wade through at times (with some examples of marginally correct grammar), but the ideas the book expresses are excellent.
- Many of the patterns for software systems were presented in workshops at the Pattern Languages of Programs conferences. A good collection of the patterns from these conferences appears in the *Pattern Languages of*

Program Design series (Coplien and Schmidt 1995; Vlissides et al. 1996; Martin et al. 1998; Harrison et al. 2000).

- The Hillside Group's patterns page is a good starting point for all things relating to software patterns, including information on the various patterns conferences. The URL is http://www.hillside.net/patterns.
- For examples of patterns applied to software systems, *Design Patterns* (Gamma et al. 1995) is the one book "everyone" has read. The *Pattern-Oriented Software Architecture* series of books is another example. These books do not approach the richness of the Alexander books, but they provide a concrete example.
- Brandon Goldfedder's *The Joy of Patterns* (Goldfedder 2002) is an excellent introduction to using the design patterns.
- *The Manager Pool: Patterns for Radical Leadership* (Olson and Stimmel 2002) is a collection of patterns on managing software organizations.
- *The Patterns Almanac* (Rising 2000) is one of the most comprehensive indexes of the state of software patterns in publication.

Part II

The Patterns