



**Free Software Configuration
Management (SCM) –
Is it worth it?**

**Considerations When Comparing Open-
Source SCM to Commercial SCM Solutions**

Authored by:

David Kelly and Heather Ashton

*Upside Research, Inc.
www.upsideresearch.com*

Contents

□ **Executive Summary**

The market for software configuration management (SCM) tools is filled with an array of products and solutions. Before selecting an SCM tool, it is important to understand exactly what the impact of that product can have on your development environment. This paper assesses the potential impact of open-source and commercial SCM tools in five selected areas of evaluation.

□ **Methodology and Definitions for the Report**

This paper is based on research conducted by Upside Research with software vendors and multiple enterprise developers and managers responsible for mission-critical source code.

□ **When Open Source Makes Sense and When It Doesn't**

The important thing to remember when using open-source tools for a development effort is that if the scope of the project creeps beyond the initial outline, or the project continues to grow and morph into a much larger project (as many development efforts have been known to do) there are a number of issues with the development process and the SCM tools that support it that must be considered.

□ **A Typical Scenario for SCM Tools: The Real Costs and Benefits**

This section is designed to help companies evaluate the true cost of an SCM solution. It takes a closer look at the experience of a consumer services company in the travel sector with both open-source and commercial SCM solutions, and the costs and benefits the retailer experienced.

□ **Conclusion**

Effectively managing the development effort is critical for the ultimate success of any project. Having the right toolset to support this effort is often the key to maximizing development resources. SCM tools play a critical role in this toolset and determining the best fit for your development process is an important decision to make. While open-source SCM tools are readily available and free for the taking, it is important to understand that they may not be free for your organization in the long run.

Copyright 2006, Upside Research, Inc., which is solely responsible for its content. All rights reserved. No part of this report may be reproduced or stored in a retrieval system or transmitted in any form or by any means, without prior written permission. Disclosure statement: This report was sponsored by, and may be licensed to, a limited number of commercial software vendors. No quotes or information can be excerpted or used without explicit written permission from Upside Research, Inc.

Executive Summary

Software Configuration Management (SCM) is a critical core infrastructure tool for all software development organizations. The right SCM solution helps organizations by helping them:

- improve product quality and customer satisfaction;
- meet product time-to-market and revenue goals;
- comply with government and industry regulatory requirements;
- protect key corporate assets;
- manage increasingly complex and evolving development processes, and much more.

While often taken for granted or below senior management's radar, SCM is an extremely important component of an organization's software infrastructure.

Historically, a significant number of organizations have used open-source SCM, such as RCS, CVS and more recently Subversion, to form the backbone of their development environments. However, such solutions can have hidden costs associated with them. This paper was written to help organizations understand when such free tools make business sense and when commercial SCM solutions may be more appropriate. We wanted to ask the questions that senior management should be asking themselves:

- How do I know if open-source SCM is right for my organization?
- What risks am I exposing my organization to by using open-source SCM?
- What are the true hidden costs of open-source SCM?

In this report we provide context for answering those questions and analyze a composite scenario of an organization that has used both types of solutions, based on interviews with enterprise developers and industry research.

While our findings show that open-source SCM is undeniably a valid option for certain companies and certain types of projects, it can also be a very expensive option once all the hidden costs and limitations are added in. In addition, open-source SCM can also harbor hidden risks and costs that may be considered unacceptable to many organizations given the demands of today's global and compliance-driven business environment. Such risks may include: the need for time-intensive (and potentially error-prone) scripting required for proper implementation; the exposure of having someone new assume the management of the tool when turnover occurs; and the compliance failure resulting from the inadvertent and the malicious deletion of history/artifacts in the tool (deleting history is a practice which many open-source SCM tools allow). Organizations considering open-source SCM solutions to meet tactical or strategic project needs should pause and consider the implications of such solutions on the long-term efficiency and effectiveness of their development processes.

Of course, at the most basic level, commercial SCM products cost more initially, at least from a purchasing standpoint. There are also on-going hard costs (such as maintenance

fees) and soft costs (such as administration or management resources) that need to be factored into a purchase vs. open source decision.

Upside Research knows that the market for SCM tools is filled with a wide range of both open-source and commercial solutions. This paper assesses the potential impact of open-source and commercial SCM tools on five critical areas of evaluation in order to provide greater insight into the potential costs and/or benefits associated with any given solution. Upside Research suggests that development teams and managers use this document as a starting point for determining the best SCM tool fit for their requirements.

METHODOLOGY AND DEFINITIONS FOR THE REPORT

This paper is based on research conducted by Upside Research with multiple enterprise developers and managers responsible for mission-critical source code and software vendors. The paper is intended to highlight a select number of issues that organizations need to consider when evaluating either open-source or commercial SCM packages and help organizations understand the associated potential costs or benefits. We suggest that readers use this report as a basis for constructing their own case for investment in a specific commercial package or as the justification behind why open-source or freeware SCM will meet both current and future needs. Within large and small development organizations, this report will benefit release engineers, senior developers, SCM leads, project leaders, managers, architects, and anyone that needs to evaluate solutions for long-term viability and not just for single, one-off projects.

For the purposes of this report, we are calling open-source and free solutions (SCCS, RCS, CVS, Subversion) "open-source SCM" and commercial solutions (AccuRev, Borland, IBM Rational, Serena, Telelogic, etc.) "commercial SCM."

WHEN OPEN SOURCE MAKES SENSE AND WHEN IT DOES NOT

Open-source SCM tools are often a preferable option for organizations in certain situations. When a development project involves a small team and is limited in scope, most commercial SCM tools are overkill. It would take just as long to finish the development of the new project as it would to get up and running with the commercial SCM tool. Therefore, using an open-source SCM tool in this case makes perfect sense. Other factors that point toward open source include either limited (or no) parallel development or a team of developers that are all physically located together. Upside Research has spoken with several large enterprises that have validated the successful use of open-source SCM tools in these types of situations with good results. In almost every case, these

When Open Source May Not Be Enough

How well does the open-source SCM tool support these requirements?

- Advanced parallel development environments
- Controlled distributed development
- The need for atomic saves
- More than ten-person development projects sharing common code
- Mission-critical or commercial software development
- Multiple branches and merges
- Complex repository structure
- Need for reproducibility
- Regulatory compliance requirements

companies also had licenses for commercial SCM products, but found that for the fast and straightforward development projects, using an open-source tool proved more efficient and cost effective. To help provide context, Upside Research urges companies to consider the following issues related to SCM deployment:

Scope – When using SCM products it's particularly important to consider the potential for the scope of a project to creep beyond the initial requirements and the potential for the project itself to grow and morph into a larger project (as many development efforts have been known to do). Increasing the complexity of or size of a project can have a dramatic effect on the true cost and resources required to make an open-source SCM solution work efficiently. There are large enterprises that have found themselves years after adopting a "free" open-source SCM tool still trying to make it work. And, while technically such tools will work, the associated costs of having to make them fit into a team of tens or hundreds of developers when once there were five may be simply impractical.

Responsibility - Who will manage the scripting/customization and upgrades for the open-source tool? When a commercial tool is purchased, it often has direct ownership tied to it for the purposes of upgrades, maintenance, and management. However, the responsibility for an open-source tool often falls in the lap of whoever initially downloaded the product and brought it to the development team. If the project has grown and now the open-source SCM tool has been in use for enough time that there are upgrades available, that same developer who brought the tool to the table now must manage and administer it, which invariably means less time spent on actual development. The task of managing both the tool and the associated scripting can be especially daunting as the organization scales.

Scripting - The common thread in our research is that the costs of customizing the open-source SCM tool are often drastically underestimated when the team first begins using it. Where does the tool stop and the scripting begin? Often teams find themselves up to their elbows in scripting when they assumed at the outset that the customization needed would be minimal. Another common and underestimated outcome is that the development organization ends up modifying its development process to conform to the limitations of the open-source SCM tool, especially with respect to supporting branching and merging operations, which often causes non-trivial costs related to reduced developer productivity and/or software quality.

Compliance - Compliance is another area that must be considered when making a decision about open-source SCM. How much risk are you willing to allow into your development environment? For companies that have classified development efforts, choosing an open-source solution may open the door for levels of risk that are simply unacceptable. Because of the nature of open-source, there is no protection from some malicious entity contributing to the open source code and potentially causing significant harm. While not a day-to-day problem, the level of acceptable risk from an open-source solution is something that needs to be weighed against the benefits of using an open-source tool.

The above issues are often overlooked by many development groups when looking at the relative simplicity and convenience of an open-source SCM tool when compared to a commercial package. Only after using the tool for a while, when it is considered part of the fabric of the development project, does it suddenly cross over into the category of overhead burden and administrative challenge.

Keep in mind that the considerations we raise in this report must be put into context within your specific deployment scenario. For example, open-source SCM solutions may work well for even large companies with 100 or 200 developers, if their use fits the constraints of the tools—such as if the development organization is organized into 20 smaller teams that don't work with overlapping source code.

To assist organizations in evaluating whether they are at the point where an alternative to open source makes more sense, Upside Research has taken a closer look at several major areas of SCM tool functionality to help organizations in their evaluation process.

A TYPICAL SCENARIO FOR SCM TOOLS: THE REAL COSTS/BENEFITS

To help organizations evaluate the true cost of an SCM solution, Upside Research spoke with a number of enterprises currently using open-source and commercial SCM tools. We learned first-hand about the real costs and benefits realized by these organizations. The following represents a typical scenario of how one company fared, offering examples of the major categories for consideration when adopting an SCM tool.

The company is a major consumer services provider in the travel sector whose business runs on its web-based software applications. Because the code *is* the business, developers and the management of the development process are central to this organization's success. Over the course of several years, the company grew from fewer than 50 to more than 300 developers, and was hitting the wall with its open-source SCM tool with the following costs/challenges:

- **Branching and merging.** When the open-source tool was first introduced, there were a manageable number of branches required to meet the needs of the development environment. However, at its height, there were more than 200 branches off the main code branch. At that point, the open-source tool could not keep up with the merges that were taking place and as a result, two developers had to be redirected for two weeks straight to do a manual merge, at a cost of more than \$12,000 for a major merge tied to a code release.

The Human Risk: An Employee Turnover Example

The developer that "owned" the tool was able to make minor fixes or enhancements in a few days or within a week. He also had intimate knowledge about how to fix things when there were problems. When he left the company, there was a huge hole in his absence. Suddenly, those minor fixes were taking weeks or months. The tool needed to be upgraded, and it was a "complete nightmare" for the new developer, who had to try to migrate all of the customizations without understanding their genesis.

- **Auditing.** At the beginning, when the development team was small and the process fairly straightforward, no one gave much thought to merge history and auditing. However, as the complexity grew, it became increasingly important to track merges to determine what had been merged, and which developers were responsible for what code. This was an attempt to stay on top of the complexity and ensure that merges were successful. The development team found itself wasting up to three weeks researching audit requests, which put another significant dent in productivity.
- **Change sets.** The open-source SCM tool that this company was using didn't support change sets. As a result, the developers were working overtime trying to track down patches and ensuring that they were made in all the necessary areas. The costs added up to more than \$10,000 annually.
- **Global view of project.** Having a global view (or graphical view) of the development efforts was seen as important in order to optimize the performance of the team. Without this, this company was seriously challenged by the 200 active branches within the project. Developers had little insight into the larger project picture which led to the habit of "throwing it over the wall" with their current work. This proved to be a primary reason why the company decided it needed a commercial SCM tool.
- **Distributed development.** As the company grew, it outsourced development to more cost-effective geographic locations in order to reduce development expenses. This meant that there were now developers located across the globe, contributing to a common source base 24/7. The open-source SCM tool that they were using did not effectively support distributed development. The code review process was challenging as a result, and often the developers had to resort to time-intensive brute force methods to manage off-shore contributions, where a developer that was intimately familiar with the structure physically checked in code to the intended branch.
- **Vendor support and upgrade path.** While in the beginning there wasn't much overhead associated with managing the open-source tool, the development team found that this changed significantly with time. The developer who initially brought in the tool found that over four years he had spent as much as 50% of his time doing custom integrations and upgrades. There were few, if any, outside sources of support that he could tap into, and this significantly curtailed his productivity.

The Real Costs of Open-source SCM For This Development Organization	
Category	Cost
Merging	More than \$12,000 per major merge = \$60,000 annually
Auditing	3 weeks of developer time doing audit research = \$9,000
Change sets	More than \$10,000 annually tracking down patches
Vendor support and upgrades	\$40,000 annually for one developer to spend half his time managing integrations and upgrades
Total costs*	\$119,000 per year to use the open-source SCM tool

THE COMMERCIAL SOLUTION AND ITS BENEFITS

This company chose a commercial SCM tool to replace its open-source tool, and immediately saw a considerable gain in productivity and cost savings. There were a number of areas where the new tool proved beneficial and made positive contributions to the development effort. Here's a closer look at the benefits:

- **Continuous merging.** With the new tool, the development team began to adapt to a continuous approach to development, where they were continually merging changes. This prevented errors from making it through to the end of code development and throwing the team back weeks.
- **Audit history.** Shortly after implementing the new SCM tool, the development team was able to test its audit capabilities. It had an audit request, and using the new tool was able to complete it in 2 hours, instead of the three weeks it used to take.
- **Graphical view.** The developers really appreciated the new global view of the project. This was especially critical for new hires, who were able to use the global view of the project to get up to speed quickly, saving an average of two weeks training per developer.
- **Distributed development.** With the new tool, the team was able to use an existing developer located in India instead of hiring a new developer in the US for \$80,000 annually.
- **Custom integrations and upgrades.** The commercial tool provided full vendor support and a number of pre-existing integrations, saving 50% of the time the lead developer spent managing the open-source tool.

* These are the immediately identified costs that were eliminated when the company moved to the commercial SCM tool. The actual total costs, which involved set-up, scripting, upgrading/maintenance, training, support, etc. were considerably higher.

The Real Benefits of Commercial SCM for a Major Consumer Services Provider	
Category	Benefit
Merging	Saved up to \$60,000 annually from continuous merging
Auditing	Saved \$8,900 per audit request, and provided reassurance of code safety, eliminating risks of losing work
Graphical view	Saved \$5,000 per new hire in productivity gains
Distributed development	Saved \$80,000 from having to hire new software engineer in its US office
Upgrades and vendors support	Saved \$40,000+ through vendor support, pre-existing integrations and easier upgrades
Identified Benefits	Saved \$200,000+ in first year of use

CONCLUSION

Managing the development effort efficiently is critical for the ultimate success of the project. Having the right toolset to assist with development can be the key to maximizing development resources. SCM tools play a critical role and determining the best fit for your project is an important decision to make. While open-source SCM tools are readily available and free for the initial taking, it is important to understand that they may not be free for your organization in the long run. On the other hand, commercial SCM tools require upfront and on-going investments.

A Final Glance at Selected Business Decision Points For Choosing an SCM Solution	
Scenario/Need	Likely Best Fit
Fixed-scope project that involves a limited number of developers (such as 10 or 20)	Open-source SCM
Available developer resources to customize and extend open-source projects, both initially and on-going	Open-source SCM
Mission-critical software development project that will significantly impact business	Commercial SCM
Compliance and security requirements, including the need for audit trails	Commercial SCM
Distributed development team that spans geographic locations	Commercial SCM

When a development project crosses the invisible line from simple and straightforward into distributed, parallel development of multiple branches, an open-source SCM tool may not be the most efficient solution, and developers can be left with the challenge of managing a tool that cannot easily fill the requirements. Therefore, Upside Research recommends that development teams take a close look at the projects they are launching and make a smart up-front decision about whether open-source SCM is enough or if they should instead consider a more robust commercial solution that supports parallel development and multiple branches. In some cases, failure to make that assessment at the beginning can cost a team significantly in time and resources and can be prevented with the right up-front planning. Using the categories we reviewed above, we recommend that you take a close look at your next project and determine the type of SCM tool that fits it best for both today's environment as well as tomorrow's environment.

About Upside Research, Inc.

Upside Research is a research and consulting firm focused on helping clients put application development, business process management, integration, and enterprise infrastructure challenges in perspective. Upside Research helps organizations find practical ways to achieve their IT goals and profit from the diversity of a changing technology landscape.

www.upsideresearch.com

info@upsideresearch.com