A technical discussion on Improving Business Value with Software Configuration Management
August, 2003

Rational. software

Better Software Configuration Management Means Better Business: The Seven Keys to Improving Business Value

Tom Milligan

Software configuration management (SCM) is the unsung hero of software development.

Why? First off, operating at peak efficiency, SCM solutions are scarcely seen. They should be transparent to the user, freeing developers to code without unwieldy processes. Second, rarely does anyone on the team notice SCM unless it is unnecessarily intrusive or it is broke. So the better it performs, the less you hear about it, or appreciate it.

But the reason SCM is a hero is because good software configuration management yields a positive return on investment (ROI). It will, in effect, make you money by enhancing team productivity and protecting your project assets against disasters—both small and large.

Yes, it dwells in the shadows in relative obscurity, but project managers and CIOs are starting to notice it now. They are realizing the business value good SCM offers to projects and companies.

How does SCM translate to business value?

- *Faster development means faster time to market*
- *Better quality means reduced time working on fixes rather than enhancements*
- *Greater reliability translates to more up-time and increased productivity*

ROI studies offer compelling evidence that properly configured software configuration management solutions can create efficiencies through processes and controls. These efficiencies will reduce manual tasks and save you countless hours on projects. And the more you can optimize project execution, the more business value you bring to the company.

Given the complexity of development these days, there are a myriad of ways to enhance your development efforts through SCM. However, I have simplified it to just seven attributes that a good SCM system should possess. Once understood and managed properly, these seven attributes can greatly impact your bottom line.

The attributes are:

1. Safety
2. Stability
3. Control
4. Auditability
5. Reproducibility
6. Traceability
7. Scalability

Both in my previous jobs as a release engineer and SCM consultant, and in my current job, I advise customers on how to get the most from software configuration management (SCM). These seven important SCM capabilities—safety, stability, control, auditability, reproducibility, traceability, and scalability—are critical requirements for successful software configuration management. So lets explore them a little more closely.

SAFETY

The primary purpose of any SCM system is to keep project assets (e.g., design models, source code, test cases, documentation, etc.) safe from corruption, unintentional damage, unauthorized access, or even a disaster. This requires two things:

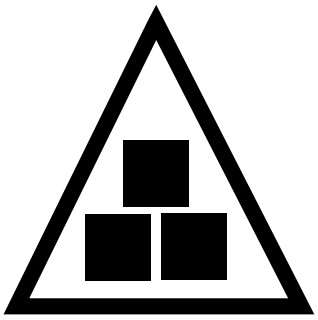
> ***Secure access***–*Those who can view or change project assets must be only those persons explicitly authorized to do so.*
>
> ***Reliable recovery***– *The ability to recover work lost in the event of an authorized user's mistake, such as the accidental deletion or overwriting of source code files.*

You cannot underestimate the business value of SCM safety features. Safety features are the basis for key areas of risk mitigation in the software development process. If work isn't secure from deliberate or accidental loss, an unacceptable level of risk to code and other critical project assets is ever-present. This potential loss can temporarily cripple a project and, at worst, derail a project for months or kill it altogether.

As an example of this, many SCM systems don't provide an easy way to rematerialize a past configuration. This forces diligent development teams to rely on other methods for achieving this capability such as writing specific configurations to tape or other backup media at critical project milestones.

This, however, doesn't protect a project from someone accidentally restoring a past configuration over the top of existing work. And it certainly doesn't allow for rematerializing configurations that don't correspond to those critical project milestones.

Business Value:
Creating safety in your development environment means having the ability keep unauthorized users out and being able to quickly restore code that gets corrupted or overwritten. In short, it is the protection of key business assets. There is time saved when you do not have to manually recreate specific configurations of your software system and can pull it directly from a repository.



STABILITY

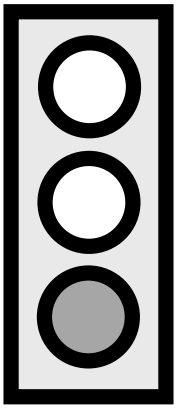
A stable development environment, both for the team and for individual developers, is indispensable. True stability has two essential elements:

Guaranteed stable work areas– *Many SCM systems can destabilize individual work areas when others check in new code. Developers should be able to leave work and return the next day–or at any future point–knowing the data on their desktops has not been altered without their consent.*

Individual control over when and what new code (which may not be the most stable) is introduced into a work area–*For example, a developer who has been working on his own for several weeks should first have enough control over his environment to decide when he is ready to introduce new code (and potential instability) into his environment as well as the team's environment.*

Beyond that, the developer should also have the option to update his environment gradually—to evaluate stability levels. The alternative is to do it all at once and potentially introduce widespread instability into the developers workspace and the project. This level of control – the ability to select when and what to introduce to individual workspaces – significantly reduces project risk.

Business Value:
When you introduce instability into a developer's environment, it can cause a downward spiral, whereby developer and team productivity dramatically suffer. This will have a negative impact on morale, cause schedule delays and quality problems. Maintaining a stable environment negates these problems and adds value across the board.



CONTROL

A major role of any SCM system is to help manage change throughout the development lifecycle. The system must strike a balance between enforcing appropriate controls on the overall workflow of a project without imposing bothersome constraints on individual team members.

Today developers are often working from different offices, in different countries, and in different time zones. Trying to integrate contributions from many diverse development teams requires a system that can control:

- *Who works on a change request*
- *How changes flow from development to integration*
- *Who can work on a particular development stream*

On the flip side, your SCM solution needs to be flexible enough to allow, perhaps a whole team, to work on a common branch of code, or allow individual members of the team to work on "private" branches. In the case where private branches are desired, then the system needs to have the ability to control the flow of changes between those private branches and the projects integration area.

Today software re-use is important when it comes to reducing costs. Therefore, it is invaluable to have the ability to enforce a "food chain" approach to development. This approach is when a team produces deliverables that are meant to be consumed by another team on the project. Such components should be modifiable only by the team that produces them.

Properly understood and implemented, these features create a controlled environment for development, which results in more productive developers and more predictable project schedules.

Business Value:
Software reuse is essential to reducing costs. And you need to set policy to achieve this and other goals. Control in this context is about planning how you want to work and establishing appropriate enforcement of those rules. Consider how many great accomplishments just happen without a plan, a process, or a roadmap for success. Not many.

Control may appear to be slowing progress, but it is really just creating order to better achieve a specified result. It is analogous to deciding to walk on the sidewalk. If we all agree to walk on the sidewalk and let trucks drive on the street, then in return we don't get hit by the trucks. This kind of planning and control is good business.

AUDITABILITY

Developing software is an iterative, complex process that requires a keen eye to understand all that is happening. It is similar to our perception of a grandfather clock. On the outside, the hands move, chimes gong, and it keeps track of time. But on the inside there is constant motion. Gears interlock, churning at different rates. Special cogs are calibrated to the milli-second interlocking with others. All the parts are balanced and aligned to achieve absolute precision— a harmony that looks so simple from the outside.

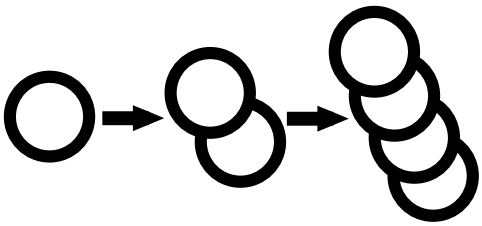
Software presents the same paradigm with one exception—it is exponentially more complex when you look under the hood. And amid all the activity, you sometimes have to ask whodunit and why.

This is where auditability comes in. Auditability refers to the ability to answer the who, what, when, and where questions about a specific version or configuration of the software release at any time. It also must be able to highlight the differences between releases. Build engineers, managers, and developers must have ready answers to common queries such as:

- *Did component foo.y get modified according to latest requirement?*
- *Who made a change to line 10249 of version 3.2.4?*
- *Were all the top priority bugs fixed for the patch release and who checked them in?*

The answers are critical for smoothly resolving familiar occurrences such as errors in build scripts or include files. Finding the information manually consumes valuable time. Indeed, developers frustrated by an inability to interrogate the SCM system for this critical information may simply let build and testing teams sort things out—wasting even more time and greatly diminishing project efficiency.

Business Value:
Auditability requires that an SCM system track and record metadata (data about data) about changes. Just as importantly, it also allows you to query the database to find your answers easily. Auditablity can reduce the time looking for answers from days to seconds. Multiply that by a half dozen questions per day, and you'll realize a heap of business value!



REPRODUCIBILITY

Among the many changes made over the course of a project, not all are for the better. Sometimes, previous versions are superior to the latest. You must be able to quickly reproduce that previous configuration to get the project back on track.

Reproducibility is like an aerial snapshot. It represents a distinct time and a detailed, broad view of the project.

Reproducibility in the context of software development requires:

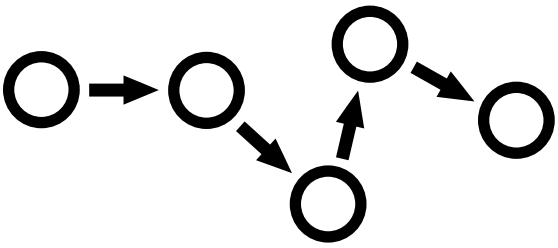
- *The ability to reproduce the configuration of the entire development environment, including tools, tests and documentation at any point in the project lifecycle. This is crucial, yet few products deliver fully in this regard.*
- *The ability to automatically recreate not only the files used in a build, but also the directory structure and the entire name space.*

Very often the directory structure of an earlier release is reorganized— file or sub-directory names change, large directories are sub-divided into smaller ones and files are moved around. The ability to re-create the prior release directory structure is called name space versioning. And SCM must provide a mechanism for name space versioning, not just file versioning.

For example, it may be useful to roll back the code base to a point in time that wasn't particularly notable, but which embodied a better version of code. Rollback requires reproducibility. And depending on the scope of the rollback, it may require a tremendous amount of detail that only name space versioning can provide.

Business Value:
Reproducibility in terms of rolling a project back to a milestone or an arbitrary point in time may only yield periodic value. But when used routinely to ensure that what is being built also has been tested, its risk reduction value is incalculable.

Reproducibility is like version control on steroids— permitting you to reproduce exact file versions and configurations of every file in a build. What might take a team days or weeks to reconstruct can happen in seconds.



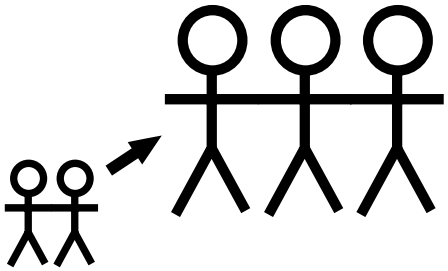
TRACEABILITY

Many SCM systems offer some degree of traceability. A comprehensive definition of traceability involves the ability to:

- *Identify the version of a released product deployed (at a customer site), in a way that makes it possible for the SCM system to easily re-create the development environment (code, documentation, use cases, etc.) required to rebuild, test, and/or run that version.*
- *Trace backward from a specific software version to the change request and/or requirements that were implemented to realize that version. Incidentally, this capability is now mandated on projects involving government agencies, such as the FDA, FAA, or DoD, all of which build strict traceability requirements into their bid requests.*

Armed with traceability features, support engineers have all the information they require to re-create the exact configuration of any customer system, and to answer the question: "Why does the software work that way?"

Business Value:
Similar to auditability, traceability is a way of finding out how you arrived at where you are in a project. Given the intricate complexities of software development and the high risks involved, traceability is essential. Like auditability, traceability saves time, recording metadata that would otherwise need to done manually or probably not at all.



SCALABILITY

As the foundation for the entire development platform, the SCM system must support projects of any scope. That is, it must scale to support large teams from start to finish, but it must not impose a burden on small teams.

A scalable SCM system should be:

- ***Configurable and functional when few controls are necessary.*** *Team members will otherwise spend undue time wrestling with an overly complex SCM process without benefit. Small projects cannot afford burdensome administration.*
- ***Versatile to manage growth.*** *SCM scalability is of increasing importance when small projects are successful. Small projects often spawn large projects, requiring advance functionality like parallel development. A SCM solution needs to be able to manage this growth.*
- ***Able to support geographically distributed teams, telecommuters, and/or outsourced team members.*** *The presence of off-site contributors adds significant stress to the SCM environment, because of the need to coordinate and manage distributed collaboration, whether centrally or via replication. Scalability also implies that reasonable performance standards should be achieved. Increasing demands on an SCM system should not compromise reliability or impede basic operations.*

Business Value:
Imagine having to move your business to larger quarters every time you hired another employee. Now imagine having to buy a new SCM solution every time you started a bigger project. It would consume time, money and cause a fair amount of aggravation and discomfort. Your SCM solution impacts too many people for it not to be scalable.

Good SCM is Good Business

A robust SCM system creates a secure and predictable environment for working with project assets. It makes it easy for individual team members to adhere to established processes and "do the right things," while making it hard to do the wrong things.

Effective SCM will:

- *Provide development management with key status information and data.*
- *Automate everyday build and versioning tasks, and provide quick access to file and version information.*
- *Support end-to-end tracking of defects and enhancements against previous file versions.*
- *Be agile and robust enough to easily adapt controls to changing project conditions.*
- *Make it is easy to do the right things and hard to do the wrong things.*

The sum of all these benefits amounts to more effective software project management— reducing bottom line costs and enhancing business value.

Over the years, I have helped many companies with their software configuration solutions. I've seen the awesome and the awful– and probably just about everything in between.

But regardless of the situation, once the customer and I came to an understanding of how an SCM solution can help their software development (the seven SCM attributes), we were able to implement a plan to improve productivity. No two plans are alike, because no two businesses are alike. Yet these seven attributes serve as a context for conversation about SCM practices. Once leveraged, these best practices will transform the way your team builds software and positively impact your overall productivity.

The business value of SCM is proven– anecdotally and empirically. Good SCM can make a great difference. You can bank on it!

Today SCM is the unsung hero of software development. But before long, as more and more project managers and CIOs sing its praises, it will soon be the MVP (most valuable product).

IBM software integrated solutions

IBM Rational supports a wealth of other offerings from IBM software. IBM software solutions can give you the power to achieve your priority business and IT goals.

- *DB2® oftware helps you leverage information with solutions for data enablement, data management, and data distribution.*
- *Lotus® software helps your staff be productive with solutions for authoring, managing, communicating, and sharing knowledge.*
- *Tivoli® software helps you manage the technology that runs your e-business infrastructure.*
- *WebSphere® software helps you extend your existing business-critical processes to the Web.*
- *Rational® software helps you improve your software development capability with tools, services, and best practices.*

Rational software from IBM

Rational software from IBM helps organizations create business value by improving their software development capability. The Rational software development platform integrates software engineering best practices, tools, and services. With it, organizations thrive in an on demand world by being more responsive, resilient, and focused. Rational's standards-based, cross-platform solution helps software development teams create and extend business applications, embedded systems and software products. Ninety-eight of the Fortune 100 rely on Rational tools to build better software, faster. Additional information is available at www.rational.com and www.therationaledge.com, the monthly e-zine for the Rational community.

The IBM home page on the Internet can be found at **ibm.com**

The Rational home page on the internet can be found at ibm.com/rational

e business software

TP902