



Increasing business values with efficient Software Configuration Management

A Softhouse White Paper

Leif Trulsson

February 2005

Contents

| | |
|--|-----------|
| Managing Software Assets is Good Business | 3 |
| Understanding the SCM Process..... | 4 |
| Version control..... | 4 |
| Configuration control | 5 |
| Process management..... | 6 |
| Problem and change tracking..... | 9 |
| Adressing Key Business Challenges | 10 |
| The 9 Key Business Benefits of Efficient SCM..... | 12 |
| Control | 12 |
| Tracking | 13 |
| Consistency..... | 14 |
| Reliability | 14 |
| Stability | 14 |
| Flexibility..... | 14 |
| Scalability | 15 |
| Productivity | 15 |
| Quality | 15 |
| Efficient SCM Will Increase Business Values..... | 16 |
| About Softhouse | 17 |

Managing Software Assets is Good Business

Today we rely so heavily on our software systems and applications that any disruption in service can have devastating effects on the business, and millions of dollars can be lost in a couple of hours. Many are the organizations that have received a so-called hot fix, applied it, and then paid the price. The fix was hot but not in the sense the receiving organization expected. Therefore we have to value and manage our software assets like any other assets. And managing our software assets is good business.

The importance of software asset management (SAM), and the importance of managing it over the whole lifecycle cannot be overstated. In fact, software lifecycle management (SLM) has become so crucial for any organization, whether they are developing systems and applications themselves, or they are purchasing systems and applications from software vendors.

To manage our software assets, we need the right process and the right tools. And just like you need an order entry system to manage your customer orders, or a general ledger system to manage the financial transactions and bookkeeping of your company, you need a software configuration management (SCM) tool and process to manage your software assets.

In this white paper we will look at the SCM process and how using the right SCM process and the right tools can improve business values.

Understanding the SCM Process

The goal of SCM is the control over the following areas:

- Version control
- Configuration control and delivery
- Process management
- Problem and change tracking

Version control

Version control and versioning (see Figure 1) is the making of copies of data at some meaningful point in order to return to that point at a later date, if necessary. And if the goal is to protect the software assets through the whole lifecycle, then we should version control all our business critical software assets. The key is to store any artifact that is not easily rebuilt from other artifacts.

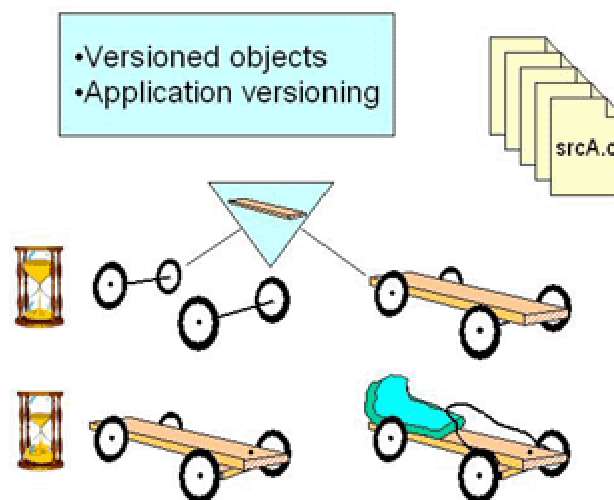


Figure 1. Versioning

Versioning can be implemented in various ways, depending on the specific tool chosen. Some tools use a full-copy versioning model, which means that they store a complete copy of the new version of the artifact being versioned. Other tools, however, use some form of delta versioning, which means that they only store the difference between succeeding versions of the artifact.

Another important aspect of the versioning model is the ability to apply clear separation of concern on various levels of the versioning. The most common method to apply separation of concern is through a branching model. It is also important to be able to apply branching on various levels of abstraction (see Figure 3).

Things you might consider storing in your source code control system are intellectual properties such as:

- Requirements documentation
- Design documents/models
- Source code
- Icons, bitmaps, and other graphics
- License files
- Readme files
- Informal development notes
- Documentation
- Help files
- Build scripts
- Database build scripts
- Test scripts
- Install programs
- Application packages (if they are not easily rebuilt)

Storing any artifact that is not easily rebuilt from other artifacts in one place makes it much easier to support and maintain the product. If and when you have to address a defect or change request, or just have to create a new CD with a specific version of the product, this is much easier done from a common repository than trying to assemble the product from disparate locations and versions of every artifact of the product.

However, once you store something in the system, you also have to manage it through the system. This means that developers (or anyone else) should not be allowed to copy or extract things to a private work area not controlled by the system. Even if their intent is to check in the artifact again, once an artifact escapes from source code control, you lose track of the artifact and the ability to easily recreate the state of your product at a certain point in time.

Configuration control

Configuration control, as said earlier, implies a higher level of abstraction than version control. The SCM tool must have some knowledge of which versions from a set of artifacts comprise a specific build.

In Figure 2 we see a typical product configuration. Product A consists of several folders, and each folder contains several versions of several files.

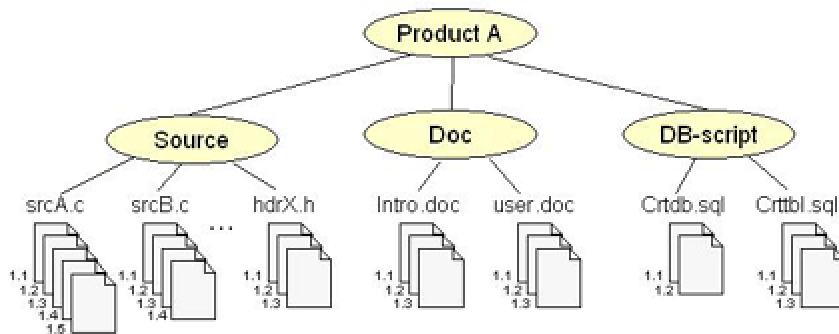


Figure 2. Product configuration

A release is a specific configuration of a product, that is, specified versions of all the artifacts that comprise this configuration. A release is a logical organization, or mapping, of all managed artifacts that are related to an application or a product. A release does not affect the physical location of a managed artifact; instead, it provides a logical view of the managed artifacts that must be built, tested, and distributed together (see Figure 3). Also, a specific artifact can be in multiple releases.

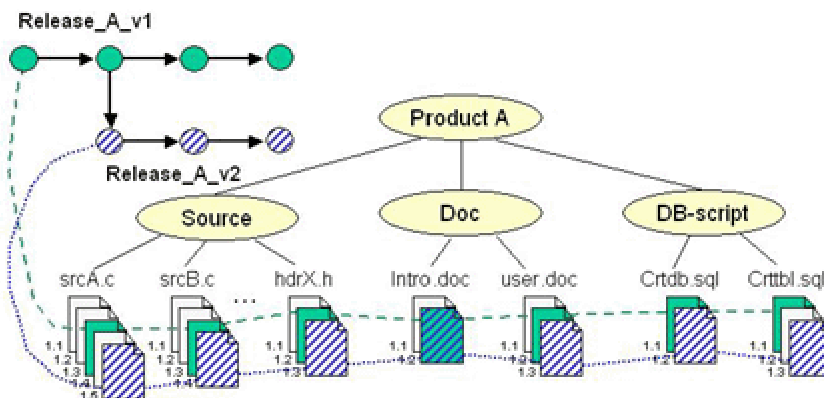


Figure 3. Release configurations

Each time a development cycle begins for the next major version of a product, a separate release is defined. Each subsequent release of a product references many of the same managed artifacts.

Process management

Process management deals with the grouping and manipulation of versions of software assets as they progress through the software life cycle. This typically involves change management, approval levels, and production control.

A process enforces a specific level of control on the various product configurations and releases. Early in a product's lifecycle we might want to apply loose control, so that versions of artifacts may be created without any approval or test process being applied. But later in the development process, we want to apply a more stringent process to ensure the quality of the product. And when the product enters the maintenance phase, we apply the most stringent process control available, so as to assure that we do not introduce any unwanted bugs or uncontrolled changes to the product (see Figure 4).

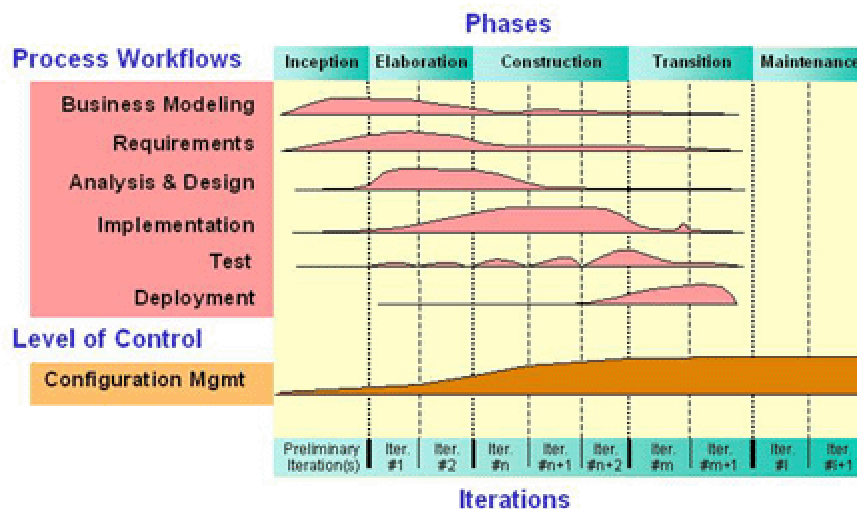


Figure 4. SCM level of control during the product lifecycle¹

Process management also involves who can do what, when, and where. In other words, we have to be able to control and define the correlations between roles, rules, and tasks.

A good SCM-tool should be able to reflect the needs of an organization and the different roles within the organization, by allowing the definition of various authority groups (see Figure 5).

¹ The figure shows the product lifecycle in relation to the Rational Unified Process (RUP)

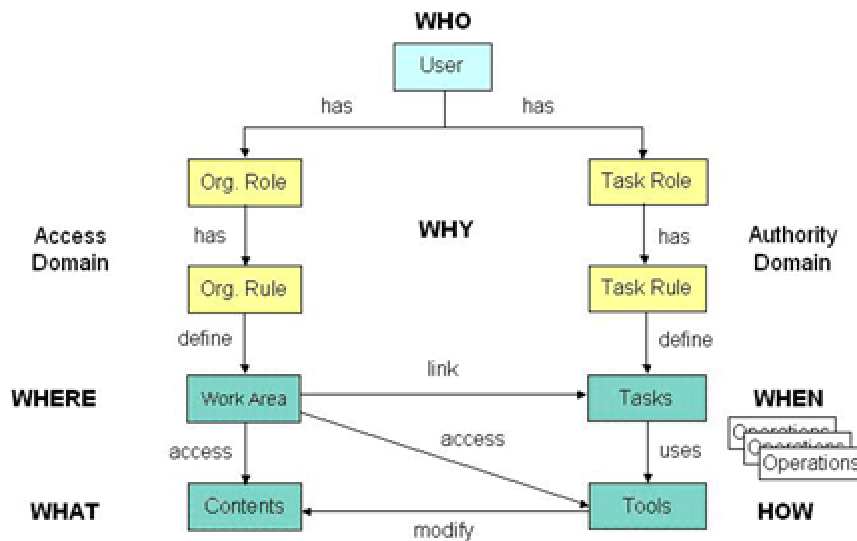


Figure 5. Relations between role, access, and authority rights

To be able to apply the right level of control, we also have to define the required states in the process workflow. A specific state determines what action can be performed on the specific object, and what state transitions can be done from there (see Figure 6).

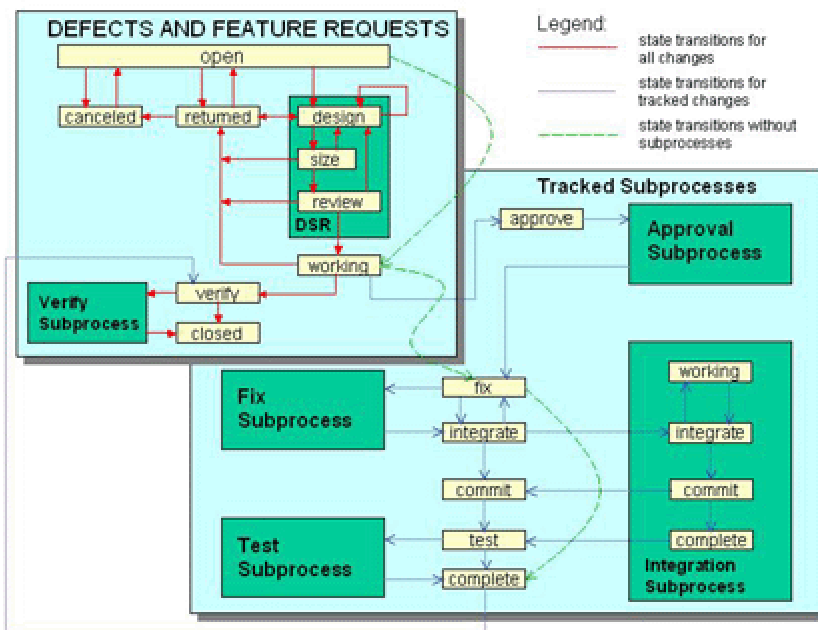


Figure 6. Example SCM state transitions

Defining the right level of control, depending on the various states in the process workflow, is key to a successful process management strategy. We want the SCM tool to support our development and support efforts, not to hamper them.

Problem and change tracking

Problem tracking entails recording enhancement/change requests or defect reports and correlating these with the resolution of the request. These reports may include a listing of the sources involved in the change. These change sets can then create released products containing only the features and fixes desired.

Problem tracking and change control enables you to manage and control the development process (Figure 6). SCM tools with integrated problem tracking allow you to track reported problems and design changes and retain information about the lifecycle of each. A defect track record is used to record each reported problem. A change request track record is used to record each proposed design change. The information recorded about defects and change requests enables you to report on the who, what, when, why, and where of modifications, as well as where a particular defect or change request is in the development cycle, and where the release is in the development cycle.

For instance, you can use the defect and change request information to answer questions such as:

- How many features have been implemented or still have to be implemented?
- How many defects are open?
- How many change requests are still in test? (Do I have to move resources to test?)
- How many defects have been opened against each managed object? (Where are my code quality problems?)

Addressing Key Business Challenges

Software configuration and change management solutions help you implement a managed approach to change that also guards against corruption of assets. Such tools should also provide support for team collaboration and parallel development, making them more productive, and leveraging geographical boundaries.

Software configuration and change management enables development teams to capture, control, and securely manage software changes and assets through the whole lifecycle (Figure 7).

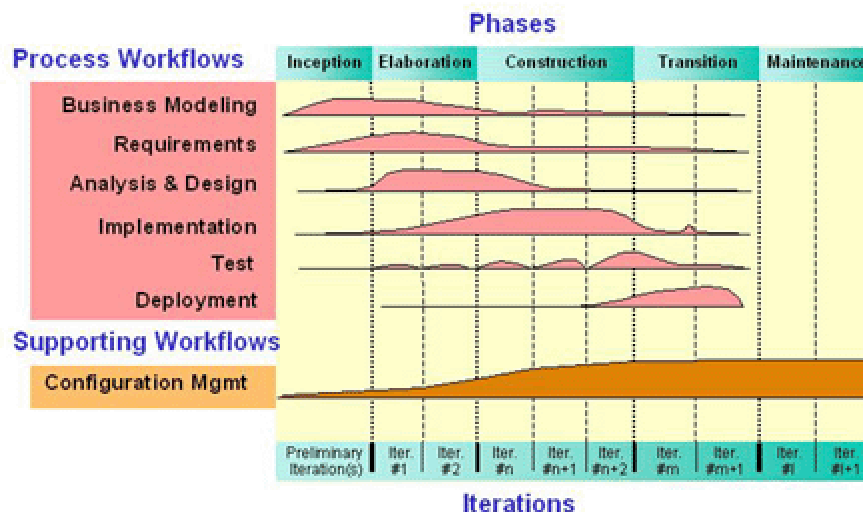


Figure 7. The software development process²

SCM solutions address key business challenges by:

- Supporting local and distributed teams where assets and changes must be seamlessly coordinated across local and dispersed development projects
- Addressing key compliance issues arising from regulatory, quality, and engineering process requirements
- Ensuring, as teams, resources, and applications change, that change management processes can quickly scale

Software configuration and change management is a key capability in modern software development practice. It allows teams to carefully track requirements over the project

² The figure shows the software development process as outlined in the Rational Unified Process (RUP), and with the addition of a maintenance phase

lifecycle, during which numerous changes — including changes to the requirements themselves — occur.

Change and asset management (Figure 8) offer the following key advantages for software development.

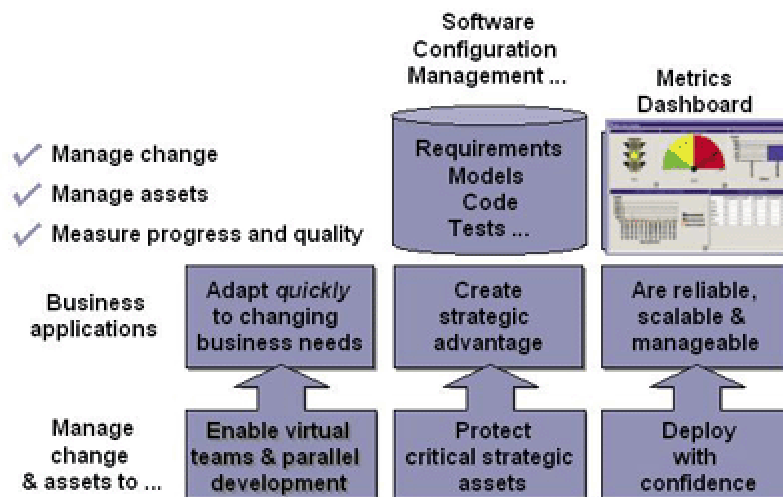


Figure 8. Manage change and assets

Enables virtual teams and parallel development — Advanced SCM systems allow multiple, sometimes overlapping, branches of a project to be worked on by different development teams simultaneously, so more work can be accomplished faster, on demand, without sacrificing quality.

Protects critical assets — A company's software development assets (requirements documents, design models, source code, automated test suites, and so forth) are unique, strategic resources that cannot be purchased or recreated from outside sources. Just as valuable as a corporation's business assets, these software development artifacts must be managed and protected. Effective change management systems ensure that no unit of code or component under development is ever lost or over-written. This affords an important safeguard against the threat of security breaches or disaster.

Allows confidence in software deployment — Change and asset management ensures that teams who are building and maintaining complex systems remain in sync as they combine multiple versions and various pieces of software code. Change management systems also allow all requirements to be tracked throughout the project lifecycle, so that the high-level architecture translates to a software system focused on user expectations.

The 9 Key Business Benefits of Efficient SCM

One of the major challenges today is to manage change and complexity. Change is inevitable, and complexity is a fact and result of the growing plethora of applications and systems that makes up the core of today's business processes. To be able to manage and control the growing need for change and the increase in complexity we need a mature SCM tool and an efficient SCM process. Using a mature SCM tool and implementing an efficient SCM process leads to a number of key business benefits. The following are nine key business benefits that can provide true return on investment (ROI):

- Control
- Tracking
- Consistency
- Reliability
- Stability
- Flexibility
- Scalability
- Productivity
- Quality

In the following sections we will explore each of the nine key business benefits.

Control

The primary purpose of SCM is control and management. A mature SCM tool and an efficient SCM process allows you to manage your development process by organizing your data, controlling changes to it, and providing reporting capability.

Efficient SCM enables you to manage and control your software assets through the whole lifecycle. It also provides you with access and authority management and control. Access