Software Configuration Management in Organizations

Ross Hoffner
Information Systems Consulting

Mary J. Granger
Professor, Information Systems
George Washington University

Abstract

This paper provides a cross sectional view of Software Configuration Management (SCM) and its role in organization that leverage Information Technology (IT). Software Configuration Management may provide the ability to properly control and manage risk within software development, thereby providing substantial benefits to an organization. Some frameworks for Software Configuration Management are reviewed and recommended for an environment that is changing rapidly and needs to be effectively managed. However, implementing successful SCM presents significant challenges for organizations, regardless of their size. It is important to define and quantify expected returns from implementation of SCM. There are multiple best practices and philosophies across different SCM tools. In order to effectively manage information technology change and growth, the selection of the appropriate tool and the implementation of the various facets of SCM must align with the organization's expectations and needs.

Keywords: Software Configuration Management, technology development, implementation, organization

Introduction

Change is an inevitable aspect of every business and introduces a level of risk to the continued success and performance of an organization. This is especially true when business processes are supported by information systems. Change originates for a variety of reasons and causes those systems to also require modification. Drivers of change can include legal or regulatory compliance, internal business process improvement, the addition or deletion of a product offering or feature, or simply the need to upgrade a supporting software version. In all of these examples, it is important for Information Systems (IS) departments to be able to guarantee a smooth transition, with minimal impact to the overall business operations and the ability to conduct business. A number of related issues coalesce around these challenges and it is the successful or unsuccessful administration of them that may be the difference between continued prosperity and failure.

Software, Change, and Configuration Management are a collection of inter-related challenges facing every organization that leverages Information Technology. It is the responsibility of both management and the teams developing the end products to ensure that change is effectively managed in a way that minimizes the potential for adverse impacts. A variety of tools and interpretations of processes are available for organizations to adopt and implement. Challenges exist in the establishment, implementation and enforcement of these processes, as they require a level of commitment at all levels across the organization. This paper defines software configuration management, identifies some of the key challenges facing organizations, examines generally accepted best practices to see how they interrelate with other software development standards, and examines some of the tools currently available to help organizations with these types of change. The goal is to provide a cross-sectional view of software configuration management in the domain of information technology management.

Software Configuration Management

Software Configuration Management (SCM) is considered a subset of configuration management (CM), which is defined in the Information Technology Infrastructure Library (ITIL) v3 [1] as "The process responsible for maintaining information about Configuration Items (CIs) required to deliver an IT Service, including their relationships. This information is managed throughout the Lifecycle of the CI. Configuration Management (CM) is part of an overall Service Asset and Configuration Management Process." A training module published by the Software Engineering Institute (SEI) [2] defines SCM as encompassing "the disciplines and techniques of initiating, evaluating, and controlling change to software products during and after the development process. It emphasizes the importance of configuration control in managing software production." The software products referred to in the SEI defintion are examples of Configuraiton Items (CIs) defined by ITIL. One can easily see the two definitions are congruent with SCM being a specific case of CM.

It is not enough to simply know the definition of Software Configuration Management. In order for management to be able to make an educated decision when deciding what products to adopt and what processes to implement to aid in the maintenance of an organization's configuration items, it must also understand its costs and benefits. By making some basic assumptions about the impact of proper or lack of proper SCM best practices and an IT professional's average salary, one can calculate a rough cost. Assuming an average IT professional's salary is $100,000 per year ($130,000 with benefits) and assuming a standard 2000 hour work year, the cost of an hour of IT professional's time is between 50 and 65 dollars. If poor software configuration management practices result in a 10% increase in the time it takes to create, implement and deploy a change in the system, then the cost can be calculated as (2000 hours x 10%) x $50 dollars per hour, resulting in a cost of $10,000 per employee per year. This cost represents only the loss in employee productivity and not the monetary penalties associated with an inability to implement change on time.

As previously identified, there are a number of reasons for implementing change within an organization, including legal or regulatory compliance. Failure to successfully implement that type of change on time could result in penalties from a governing body. Delays in

implementation can also create a missed business opportunity or a missed deadline on contract work, thereby causing a financial loss.

The ability to properly control change and manage risk within software development can provide substantial benefits to an organization. As explained in Gartner's Hype Cycle for Application Development [3] "Explicit processes reduce cost and risk, and enhance agility in response to business needs. The U.S. Sarbanes-Oxley Act and similar initiatives have caused IT auditing standards to be raised, demanding better evidence of process discipline." Gartner identifies software change and configuration management as currently existing on the slope of enlightenment with a target of reaching the plateau of productivity in the next two to five years. Therefore, though most organizations may not derive a strategic advantage from implementing SCM best practices, it does suggest that currently most organizations should be trending towards them, as they will be a necessary part of application development to effectively compete in the market place.

Organizational Challenges

Any organization that leverages information technology to support any aspect of its business process creates an exposure to the risks associated with these tools. Among the most significant of these are incomplete or incorrect functionality being deployed to the production environment, continuity of process implementation during hardware replacement, or expansion and disaster recovery. Each of these scenarios can be mitigated or exacerbated through the implementation of Software Configuration Management.

For many organizations, proper software configuration management represents a significant set of challenges. Even small organizations maintaining a multi-device, multi-application environment can quickly create a scenario in which it is imperative that SCM practices be implemented. The need of an organization to manage change against its internal configuration items represents one of two major scenarios that require the implementation of SCM. The second scenario is the need of software development companies to manage change across configuration items represented by an entire product line. Software development companies create entire product lines that exist in various states of maturity. In these scenarios, multiple products or release versions may contain different versions of the same configuration item. The implementation of a configuration database and a strong process helps mitigate the risk of regressed functionality being implemented in future releases.
The scenarios mentioned above may quickly be complicated by adding in the dimensions of application architecture and delivery method. Emerging strategies such as the Service Oriented Architecture (SOA) and Software as a Service (SaaS) present additional challenges. The increase in flexibility for creating an enterprise architecture and meeting niche business needs also increases complexity and risk which must be adequately addressed. New framework suites, such as those addressed later in this paper can help to mitigate these challenges.

In addition to the operational challenges of managing specific configuration items, organizations must also address the tactical and strategic challenges of Software Configuration Management. These may include an understanding of the strategies supported by the various framework tools, the vendor strategies for bringing the framework tools to market, and the

economics of SCM. All organizations need to understand precisely where money is being spent. Before implementing an improved SCM, it is important to quantify the expected returns. While the aforementioned development efficiency is certainly a key component, it is also beneficial to attempt to quantify the agility and flexibility for bringing software enhancements to the production environment faster, through improved and shortened development cycles that can result from SCM [4].

Best Practices

Once an organization has estimated the potential costs and weighed the identified risks, it must then decide the best approach for mitigating the risks of poor SCM practices. While every organization's risk level and tolerance for risk varies, there exists a framework of focused processes that may be adopted and incorporated with as few or as many other processes as are deemed integral to daily operations. Many software vendors offering SCM packages implement the generic underlying philosophies in manners specific to their product, creating naming conventions for their product's version of an SCM. These underlying philosophies can also be found across a number of the industry's standards documents.

There are multiple operational facets of implementing SCM in an organization. According to the *IEEE Guide to Software Configuration Management* [5] there are five main areas that include:

1. Identification of the Configuration Item (CI): at what level will the Configuration Item be defined? Is it an individual file in a program, a class, library, etc? The development of a scheme or schemes to identify Configuration Items is the first challenge. The scheme(s) must also be coupled with identification and documentation of the baseline state of the identified CI. Without establishing the baseline, it is impossible to control what is being changed.
2. Configuration Control: the process for managing how the configuration item is modified from its baseline. This may include the implementation of a Configuration Control Board (CCB), a team responsible for reviewing and approving proposed changes. The CCB responsibilities may be further divided based on authority, criticality of CI by type, life cycle phase where the change is being introduced, for example. Each of these dimensions must be clearly documented to provide all impacted parties with a single source for understanding how Configuration Control will be executed and managed.
3. Status Accounting: the formal definition of the types of documentation required to track changes to Configuration Items. These types may vary in content based on the target audience and their affiliation to the overall project or CI.
4. Auditing: procedures implemented to demonstrate that the software product or Configuration Item matches its description in the specifications. This process includes maintaining and updating functional testing documents to ensure they are aligned with the Configuration Item.
5. The Release Process: this includes all activities pertaining to the implementation of a new version of software as an aggregation of Configuration Items. It should include an itemized description of the functional changes made to the individual Configuration Items as well as resolved problems.

While the IEEE standard was published in 1987, all of these concepts still persist; it is the organization and definitions of Configuration Items that have changed. Most organizations with mature software development teams continue to adhere to the methodologies prescribed by the IEEE, as these methodologies represent a practical definition of the communication and management activities required to manage change within any system. Additionally, it is the ability to establish and repeatedly follow processes like these that allow organizations to meet the requirements for the Capability Maturity Model integrated (CMMi) maturity levels. These levels measure an organization's ability to establish and maintain a repeatable structured process.

The SCM best practice framework aligns with other software development and operations management frameworks. From an ITIL perspective, best practices are aligned, as expected, with an operational focus. The ITIL definition aligns directly with the IEEE standard and focuses on the activities necessary to measure the effectiveness of the five processes. This includes the creation and maintenance of a Configuration Management database to support the CM activities. The ITIL focus can best be outlined by its key process indicators including tracking the number of Configuration Items, the number with attribute failures, and the frequency of exceptions to audits [1]. A summary of the recommendations of the primary governing and standards bodies quickly demonstrates that the generally accepted industry best practices are supported in all of the widely used process frameworks of information technology management.

Software Configuration Tools

The IEEE operational facets identify the need for implementing control and the ITIL standards work to provide insight on how to manage this control. While the general outline of the core components of Software Configuration have varied little since the IEEE standard was published, the scope and complexity of Configuration Items has. New concepts and development strategies including Software as a Service, applications built on a Service Oriented Architecture and web based technologies all create new dimensions by which software must be managed with the same diligence of control, auditing and accountability prescribed by the IEEE [5].

In the current environment, as information technology becomes more integrated with every business process, it becomes imperative that not only the best practices help mitigate the risk of unnecessary or improper change, but also that they be conducted with the same level of rigor at an accelerated pace. By meeting these objectives, organizations are able to rely on information technology departments to deliver solutions that can help business units capitalize on opportunities. It is necessary then, to implement tools that can support both rigor and speed. Organizational needs will drive functionality demand in these tools [4]. The next section investigates trends in tool evolution as well as identifying the features of several industry leaders. Historically, large software vendors have offered suite products for SCM. Leaders of this space include IBM's ClearCase, Microsoft's Visual SourceSafe, and Serena's Polytron Version Control System. As application development and configuration management has continued to evolve, these industry leaders have begun to shift focus in support of new software development paradigms. IBM's Rational suite has moved from ClearCase to Jazz™, describing the goal of this new platform for companies to "collaborate with business analysts, architects, developers,

testers, lawyers, business stakeholders, and other subject matter experts separated by time, distance, or organization [6]." Microsoft is also taking steps in updating its core product by replacing Visual SourceSafe with Visual Studio Team System and Team Foundation Server. Much like Jazz™, the Team Foundation Server includes functionality that focuses on cross-functional communication [7].

As with many other aspects of software development, there has been a proliferation of open source tools supporting Configuration Management. While a number of tools currently exist including Subversion, Git, and LibreSource Synchronizer, many of these open source tools focus on a lightweight distributed approach in which each developer is responsible for managing their own library, with resolving and merging occurring later. Additionally, tools such as Subversion and Perforce focus more on file management and change tracking rather than on the complete integrated suite of functionality defined in the commercial products. This aligns with only two of the IEEE standards, Configuration Control and Auditing, leaving organizations needing to implement processes outside of the tool to manage the other aspects of SCM. Gartner identifies the viability of these products as on their way to the trough of disillusionment with a five to ten year timeline to the plateau of productivity [8].

Software Configuration Management tools are evolving to support the changing environment of application development. Traditional framework package vendors are working to evolve their previous technologies to support new software dimensions including Software as a Service, and distributed development teams comprised of a diverse set of team members that extend beyond technical staff. The emphasis is on extending the core framework of SCM including configuration control, auditing and accounting, and the release process to bring more diverse team members to the software development process while automating many of their interactions. When selecting a product and process, organizations must carefully evaluate their needs against their expectations of the development process to ensure the best fit.

Summary

Software Configuration Management is an operational, tactical, and strategic management issue that must be addressed by all members of the IT organization. It is the processes by which change in a company's information systems infrastructure maybe effectively managed, tracked and audited. As companies continue to grow, their dependence on information technology to solve problems, automate business processes, and create strategic advantages increases. SCM becomes an imperative as evidenced by Gartner's assessment of it existing on the slope of enlightenment. Effectively implementing the various aspects of SCM and understanding the impacts of each will enable senior managers support tool implementations that most adequately align with the organization's expectations and needs of SCM.

This paper provides a cross sectional view of Software Configuration Management and its role in organizations that leverage information technology. It examines the challenges facing companies as they create or mature their software development processes in a manner that aligns with standards for both code management and the overall management of the information technology infrastructure. It is clear that the IEEE best practices for SCM are directly aligned with other best practice frameworks for information systems management, including the most

current version of the Information Technology Infrastructure Library and the Capability Maturity Model integrated. Each of these frameworks addresses the topic of change from a different perspective but ultimately acknowledges that it is change that must be managed. It is clear there is a need for a strong understanding and appreciation of Software Configuration Management if an organization is to effectively manage information technology change and growth.

References

1. *ITIL Definition: Configuration Management*. Available from <http://www.knowledgetransfer.net/dictionary/ITIL/en/Configuration_Management.htm> . [10 March 2010].
2. Tomayko, JE 1990, 'Software Configuration Management', SEI Curriculum Model, SEI-CM-4-1.4 Carnegie Mellon University Software Engineering Institute.
3. Norton, D et. al 2009, 'Hype Cycle for Application Development 2009', *Gartner Research*.
4. Duggan, J & Murphy, TE 2009 'Magic Quadrant for Software Change and Configuration', *Gartner Research.*
5. Institute of Electrical and Electronics Engineers 1987, *Guide to Software Configuration Management* pp.24-33.
6. *IBM Rational Jazz Technology Platform*. Available from <http://www-01.ibm.com/software/rational/jazz/>. [14 March 2010].
7. *Team Foundation Overview*. Available from <http://msdn.microsoft.com/en-us/library/ms242904(VS.80).aspx>. [14 March 2010].
8. Driver, M. et. al. 2009, 'Hype Cycle for Open-Source Software 2009', *Gartner Research*.