CONFIGURATION MANAGEMENT

Brouse, Peggy S.
Systems Engineering and Operations Research Department, George Mason University, USA

Keywords: Audits, baseline, change control board, configuration items, configuration control, functional configuration audit, design reviews, hardware control, physical configuration audit, software control

Contents

Summary

Configuration Management is an important activity that is essential for the success of system development projects. Without configuration management, uncontrolled changes may result in systems that do not meet user needs, need countless changes, and are often not fielded.

Configuration management (CM) is defined in this article as the process established to maintain the integrity of the work products of a program throughout the system life cycle to include all system components. The processes were identified to:

- identify a baseline for the configuration of the system at planned points in the system lifecycle,
- control changes to the baselined configuration,
- ensure the traceability of the configuration, and
- produce reports that provide status of the changes to the baseline.

CM should be applied to all program hardware, software, and documentation, regardless of whether it is developed, purchased commercial off-the-shelf (COTS), or customer-furnished. CM will be ever more important to organizations not only because of the cost incurred in fixes necessitated by the lack of CM but also because of the increasing necessity for formal CM processes to meet current process standards.

1. Introduction

It is critical that Systems Engineering professionals understand and are able to practice CM. It is often the case in systems engineering and management that:

1. CM is relegated to a secondary role and should not be so neglected.
2. Too many professionals do not practice CM in an organized manner or at all.

Effective CM is an important factor in successful system development, and maintenance. In this article, CM will be reviewed in detail including definitions, processes, current practices, and tool sets in CM.

1.1 Definition of Configuration Management

Systems engineering is more than the defined life cycle it follows. In addition to a step-wise or cyclical cycle to be followed, there are a number of infrastructure processes that must be considered through the entire life cycle including Risk Management, Quality Assurance and Configuration Management (CM). CM is necessary in the support of systems development.

Configuration Management has been defined by a number of different authors, both in commercial firms and non-commercial organizations. Babich's definition is oriented to software systems and encompasses the identification, organization, and control of software modifications in a development effort. "On any team project, a certain degree of confusion is inevitable. The goal is to minimize this confusion so that more work can get done. The art of coordinating software development to minimize this particular type of confusion is called configuration management. Configuration management is the art of identifying, organizing, and controlling modifications to the software being built by a programming team. The goal is to maximize productivity by minimizing mistakes". Tichy also addresses the management of software components. He states "Software Configuration Management [SCM] is a discipline whose goal is to control changes to large software system families, through the functions of: component identification, change tracking, version selection and baselining, software manufacture, and managing simultaneous updates (team work)".

Yourdon also recognizes the need not only manage software but to expand CM activities to other system components. "CM has long been recognized as a basic principle of good software engineering. But during the past decade, with the gradual increase in analysis/design modeling techniques, and with the advent of code-generators and CASE tools that transform those models directly into executable code, CM has come to be seen as an important issue throughout the entire life cycle of a system - from the first moment of systems analysis until the last maintenance change is made the system is retired".

Freeman also expands the definition to include other components. "Clubs such as Kiwanis, organizations such as the Boy Scouts, and companies such as IBM, carefully distinguish between those that belong to them and those that don't. Further, they may have several categories of members, such as new hire, probationary employee, temporary employee, contractor or retiree. Although they may permit visitors (another category), they are careful (for reasons both good and otherwise) to control membership in the organization. Manufacturers of complex engineered products (for example, refineries, computer systems, airplanes, and machine tools), especially when the products are made up of a number of pieces of equipment, or are one (or few) of a kind, have learned that they must carefully control and keep track of elements of the product. These procedures, often called configuration management (CM), have simple and compelling justifications".

The standards organizations, such as IEEE, also present definitions for configuration management. In IEEE-Std-610-12, the definition is expanded to include reporting capabilities. It allows for the identification, documentation, control, and verification of configuration items associated with systems development and the subsequent reporting of these activities.

It has been noted by Dart that CM is not complete if it doesn’t address process management. Configuration management as it applies to process improvement efforts is addressed in other sections of this article.

Configuration management (CM) is defined in this article as the process established to maintain the integrity of the work products of a program throughout the system life cycle to include all system components. It involves

- identifying a baseline for the configuration of the system at planned points in the system lifecycle,

- controlling changes to the baselined configuration,
- ensuring the traceability of the configuration, and
- producing reports that provide status of the changes to the baseline.

CM is applied to all program hardware, software, and documentation, regardless of whether it is developed, purchased commercial off-the-shelf (COTS), or customer-furnished.

1.2 History of Configuration Management

In the missile race of the 1950s, numerous prototypes were built, but these were under documented or not documented at all. This meant that items that were unnecessary were duplicated or added, which led to high costs and mixed success. The Department of Defense (DoD) took the lead in mandating configuration management for hardware. They issued Army, Navy, Air Force (ANA) Bulletin No. 390 and 390A (1953), which described the 'engineering change proposal' or ECP. The ECP includes the proposed change and the documentation through which the change is described in detail. ANA Bulletin 391 and 391A (1956) expanded the requirement for Engineering Change Proposals to include electronics and the manufacturing of ground support equipment. ANA Bulletin No. 445 (1963): provided a uniform procedure for the submission of proposed engineering changes to the Government for approval, stressed the definition of an impact of a proposed ECP and added maintainability and reliability considerations. MIL-STD-480 (1968) was the most complete description of change control. It provided requirements for preparation, submission and processing of ECPs and added deviation and waiver processing. DoD continued to develop military and related standards. The individual services and industry (EIA, AIA, IEEE, etc.) have developed additional standards. In addition, the process industry has included CM in their standards. The Capability Maturity Model (CMM) of the Software Engineering Institute (SEI) at Carnegie Mellon Institute includes CM as a key process area (KPA) in its Level 2 assessment. This CMM model will be no longer supported as of December 2003. In an effort to reduce the cost to organizations that may need to use multiple models, the SEI has combined models into the Capability Maturity Model Integration (CMMI). Now, organizations that have an interest in both systems and software can use a single capability maturity model. CMMI is the integrated model that was created from the systems engineering, software engineering, and integrated product and process development CMM models. The CMMI model has five maturity levels; initial, managed, defined, quantitatively managed, and optimizing. A maturity level is a defined evolutionary plateau of process improvement. They are measured by achievement of specific and generic goals that apply to a predefined set of processes and areas. As organizations progressively mature, the range of expected results that can be achieved by the organization becomes more predictable. In order to get to level 2 - Managed, there are seven process areas that must be achieved. Configuration Management is one of these process areas. The CMM suites of process models have always included CM as an important area in successful Systems Engineering.

The International Standards Organization (ISO) first published its quality assurance and quality management standards in 1987, republished this in 1994 and then published an updated version in 2000. These new standards are referred to as the "ISO 9000 2000

Standards". The ISO 9000 1994 Standards are accepted for certification until December 15, 2003 therefore CM as it applies to ISO 9000 will be discussed followed by discussion of ISO9000 2000. These standards apply to all kinds of organizations in all kinds of areas. Organizations that decide to follow these standards develop processes that meet the quality requirements specified by one of the following three standards: ISO 9001, ISO 9002, or ISO 9003. ISO 9001 is a quality assurance model made up of 20 sets of quality system requirements. This model applies to organizations that design, develop, produce, install, and service products. ISO 9002 is a quality assurance model made up of 19 sets of quality system requirements. This model applies to organizations that produce, install, and service products. ISO 9003 is a quality assurance model made up of 14 sets of quality system requirements. This model applies to organizations that assure quality through final inspection and testing.

Also considered should be ISO's guideline, ISO 9004. ISO prepared ISO 9004-1 to provide information to understand quality system elements. ISO prepared 9004-2 to show organizations how to set up a quality service system. ISO prepared ISO 9004-3 in order to explain how quality assurance concepts can be applied to organizations that process materials. ISO prepared ISO 9004-4 to explain how to develop and implement a continuous quality improvement program. ISO9000 1994 requires the fundamental components of CM including Configuration Identification. It also requires traceability to the system domain. ISO9001/2/3 require inspection and test records, which are instrumental in the CM requirement for Reporting of Status The implication is the records stay with the component, and thus by integration, with the product. These three standards also handle requirements changes, design changes, and changes to the components and product through failure to conform to specification, which cover s the requirement for Configuration Change Control. To account for Configuration Auditing, the standards require record keeping be covered.

ISO9000 2000 is replacing ISO9000 1994. When comparing ISO 9001:1994 and ISO 9001:2000 it is evident that ISO has changed the 20-clause structure of the old standard to the new standard which has 5 sections. ISO reorganized the ISO 9001 standard in order to create a more logical structure, and in order to make it more compatible with the ISO 14001 environmental management standard. The provisions for Configuration Management have stayed the same in the new standards. ISO recognizes the importance of identifying and controlling the system configuration.

As stated above, Configuration management (CM) is defined in this article as the process established to maintain the integrity of the work products of a program throughout the system life cycle to include all system components. Configuration Management (CM) is considered to be the discipline used in the system development life cycle [SDLC] to identify and control components associated with the system. The primary objective of CM is to ensure that changes to these components which may include requirements, design, and software, documentation, or hardware are controlled. Changes that are made without control may result in problems that may be cascaded throughout the life cycle. For example, changes made to requirements but not documented and consequently not made to the design, may result in code being written based on old requirement specifications. The programmer will code to the original specification unaware of changes that have been made. This will result in problems during integration and unsatisfied users.

To guard against problems such as the one just illustrated, changes to all baselined systems documentation as well as system hardware, software and other components must be controlled. During a typical systems development, the user and developer will rely on system documents for monitoring progress and ensuring that the system is complying with system requirements. Traceability of the requirements throughout the life cycle is also ensured via the control of documentation. CM is also responsible for monitoring software and hardware release libraries. CM is a critical part of the infrastructure in systems development.

It should be noted that often many functions are considered to be part of CM when, in fact, they are not. These include

- Project or product management - these functions are separate from CM and are the direct responsibility of the project management. They include resource management, scheduling, cost management as well as other tasking.
- Design responsibility - although the design phase of the life cycle depends on requirements documents baselined via CM to create the design and baselines and controls design documents using a CM process, CM has no other responsibility in design.
- Product inspection - CM does not inspect for the quality of products placed under CM control. Quality control is tasking for which Quality Assurance is responsible.

1.3 Application of Configuration Management

As one would assume, the more complex the product, the more control and formal documentation is required which is done with CM tasking. Early system life cycle phases of the project can be less structured and more open to change requiring less stringent change control. In fact, as we progress in the life cycle the more expensive each change will become. Another influence regarding the amount of CM to apply is the number of disciplines and people involved in the system development. The greater the number of disciplines and personnel involved the more the need to formalize the control process. Also, if the level of risk in the project is high, change requests must be meticulously controlled. Finally, it may be a stringent requirement of the customer that detailed CM processes be followed. No matter how small the project, however, some degree of CM will be required.

2. Configuration Management within the System Lifecycle

Within the system development lifecycle, there are a number of procedures that need to be in place from the beginning of the life cycle. To successfully control change to systems, a CM process should be put in place. A standard process should be developed for the organization and used for all system development and maintenance within the organization. The CM tasks associated with the life cycle is illustrated in Figure 1.

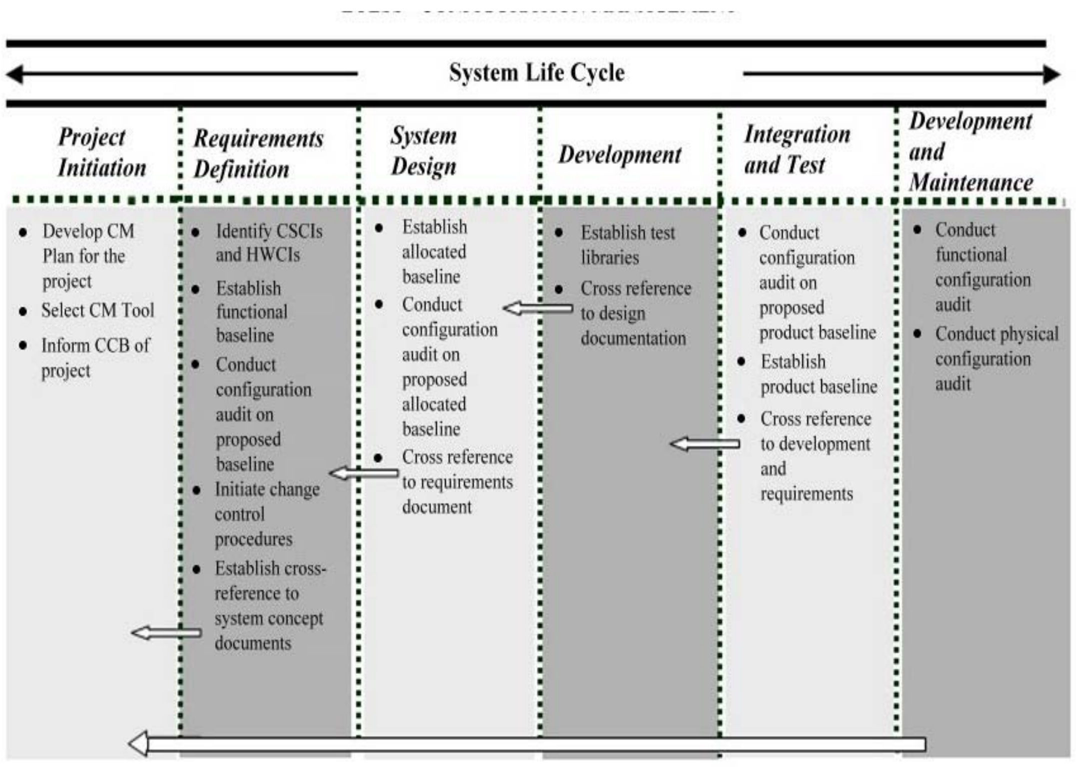
System Life Cycle
Project Initiation
Requirements Definition
System Design
Development
Integration and Test
Development and Maintenance
Develop CM Plan for the project
Select CM Tool
Inform CCB of project
Identify CSCIs and HWCIs
Establish functional baseline
Conduct configuration audit on proposed baseline
Initiate change control procedures
Establish cross-reference to system concept documents
Establish allocated baseline
Conduct configuration audit on proposed allocated baseline
Cross reference to requirements document
Establish test libraries
Cross reference to design documentation
Conduct configuration audit on proposed product baseline
Establish product baseline
Cross reference to development and requirements
Conduct functional configuration audit
Conduct physical configuration audit

Figure 1: Configuration Management Procedures

In the following sections each of the processes associated with CM as related to life cycle phases will be discussed. Many of the terms used to explain the CM tasks are generally used in the literature and accepted by CM practitioners and will be used to explain the CM process. To better understand the process, configuration management responsibilities and roles of those involved in systems development will be defined, as well as those tools that may aid in the configuration management process.

2.1 Project Initiation Phase

The Project Initiation Phase includes CM activities. Initial configuration planning is initiated no later than the inception of new contract proposal activity. Once initiated, the activity will continue throughout the lifecycle of a program in an iterative fashion. Tasks of this activity result in invocation of the other activities of the CM process.

All CM activities are planned, in detail, to coordinate both supplier and customer tasks. These plans, initiated during proposal time, are followed and maintained throughout the program life cycle and should be given status, regularly, in internal meetings. These plans include identification of:

- Products to be configuration managed and the tool set(s) to be used
- Processes to be followed and associated cost and effort for performing CM activities
- Responsibilities and schedules for initiating each baseline and for performing change control activities over those baselines

Before the requirements are gathered for subsequent development, project infrastructure steps must be undertaken. As it relates to configuration management, there are several activities that occur. As part of the planning process, the project manager should plan for and delegate responsibility for the creation of a Configuration Management Plan. The CM Plan is developed specifically for the given project, but may have been tailored from the standard CM Plan of the organization. Many organizations, both commercial and non-commercial, require compliance with a specific standard, such as IEEE/EIA 12207.0-1996 Standard for Information Technology -Software Life Cycle Processes or MIL-STD-973, Configuration Management. The project-specific CM Plan should contain the change control process as well as those procedures needed to support the process.

2.1.1 Inputs to the Project Initiation Phase

In order to successfully initiate CM activities in the project, several inputs are necessary. System development performance requirements should be made available including the list of required deliverables and associated documents including the Users Request for Proposal (RFP), the contract and Statement of Work (SOW) after contract award.

During this phase, CM cost and labor estimates are identified for the remainder of the life cycle. It is often the case that CM personnel are matriced to several projects. It is therefore important that CM personnel participate in the cost and schedule estimates that are undertaken in Project Initiation. As in any estimation effort, information is required to aid in the estimation. Examples of items that may influence CM cost are as follows:

- Milestone schedules
- Amount of travel required
- Number and type of documents that must be produced for delivery
- Number and type (i.e., source languages, operational platform, etc.) of software and hardware to be produced and controlled
- Number and type of baselines that must be established
- Number of systems to be produced, and how and where they are to be deployed

-
-
-

Bibliography

Berlack H., Berlack R. (1991). *Software Configuration Management (Wiley Series in Software Engineering Practice*. John Wiley & Sons. [Designed for software product developers, provides comprehensive coverage of the theory, practice, and techniques of good software configuration management and a structured approach to implementing these practices on large software development projects.]

Brown, W., McCormick H., Thomas, S. (1999). *Anti-Patterns and Patterns in Software Configuration Management*. John Wiley & Sons. [Patterns supply a template for the construction of software over multiple projects, while Antipatterns are commonly repeated flawed practices. The authors provide eight software configuration management Antipatterns, four management and process Antipatterns, four requirements and testing Antipatterns, and three process Patterns that address process, people, and tool issues.]

Dart, Susan A. (2000). *Configuration Management: The Missing Link in Web Engineering*. Artech House. [This practical, comprehensive book provides a thorough overview of CM technology, and reveals "best practice" techniques for selecting and deploying automated CM solutions. Nine key challenges facing e-commerce are detailed, along with practical guidelines for avoiding common pitfalls that can quickly derail your e-business.]

ISO 9000:2000 *Quality Management Systems — Fundamentals and Vocabulary*, 15 December 2000. **[**Discusses the underlying concepts and approaches for the new ISO 9000:2000 family, and provides definitions for the new vocabulary ISO 9000 is not intended as a specification, however, it is named in ISO 9001 as a normative reference and thus can be used by auditors to support their interpretation of ISO 9001 requirements - in particular in reference to the vocabulary.]

ISO 9001:2000 *Quality Management Systems — Requirements*, 15 December 2000. [Is the actual specification for the quality management system. Its requirements define the criteria for the quality system audit. The role of this standard in the series has not changed, but its content and sectional organization are completely revised.]

ISO/IEC 12207 *Information Technology - Software Life Cycle Processes*, August 1995. [This standard provides a software life cycle framework for the definition, development, and deployment of software.]

Leon, A. (2000). *A Guide to Software Configuration Management*. Artech House. [Explains how to use software configuration management (SCM) to identify the configuration of software at discrete points in time and the systematic control of changes to the identified configuration for the purpose of maintaining software integrity, traceability, and accountability throughout the software life cycle. Configuration control, auditing, documentation, and implementation are discussed. An appendix lists SCM vendors and capabilities of the tools available.]

Lyon, D. (2000). *Practical CM: Best Configuration Management Practices*. Butterworth Architecture. [Provides configuration management (CM) process implementation methodologies and guidelines for the transition from paper-based CM systems to electronic product data management (PDM) systems.]

Mikkelsen, T., Pherigo, P. (1997). *Practical Software Configuration Management: The Latenight Developer's Handbook*. Prentice Hall. [This guide to professional configuration management demonstrates how individual developers and small teams can use simple techniques, strategies and procedures for tracking and managing source code, help files and all the other elements associated with software projects.]

MIL-STD-490A *Specification Practices.* [This standard establishes the format and contents of specifications for program-peculiar configuration items, processes and materials.]

MIL-STD-498 *Software Development and Documentation.* [The purpose of this document is to establish uniform requirements for software development and documentation. EIA and IEEE have developed a commercial replacement for this standard, the U.S. implementation of ISO/IEC 12207, Information Technology - Software Life Cycle Processes.]

MIL-STD-973, *Configuration Management.* [This standard defines configuration management requirements which are to be selectively applied, as required, throughout the lifecycle of any configuration item (CI).]

MIL-STD-2549 *Configuration Management Data Interface Standard***. [**This standard defines data interface requirements for configuration management. Will replace MIL-STD-973.]

Monahan, R. (1995). *Engineering Documentation Control Practices & Procedures*. Marcel Dekker. [Sets out the requirements for establishing, maintaining, and revitalizing a system for controlling the documentation of engineering products that are used and handled by technical and manufacturing personnel in private industry.]

Sage, A. P. and Rouse, W. B. (Eds.) (1999). *Handbook of Systems Engineering and Management*. John Wiley and Sons, New York. [This comprehensive handbook contains a discussion of virtually all aspects of systems engineering and management, including a chapter on configuration management.]

Tichy, W. (1994). *Configuration Management (Trends in Software, No 2*). John Wiley & Sons [Gives a comprehensive survey of industrial SCM tools; specific integrated SCM systems such as Adele; plus novel SCM algorithms and techniques.]

Whitgift, D. *Methods and Tools for Software Configuration Management (Wiley Series in Software Engineering Practice)* John Wiley & Sons. [A comprehensive guide to the principles and practice of configuration management - the management of software system components during updating or replacement of elements. Features of commercially available tools are described enabling critical evaluation of their effectiveness.]

Biographical Sketch

Peggy S. Brouse, Ph.D. – Dr. Brouse is an Associate Professor of Systems Engineering and Operations Research and Director of the Center for Systems Engineering Technologies (CSET) at George Mason University. CSET is involved in several research areas including requirements engineering, cost modeling, and process improvement. Research clients include DoT, VDoT, DoE, VDRPT, and Motorola. Before GMU, she worked at the MITRE Corporation were she managed and participated as a systems engineer on several projects concerned with requirements analysis, conversion, testing and quality evaluation for NIST, USDA, USGS, DoL, and DoD (Navy, Army). The area of her current research encompasses process improvement, knowledge engineering, decision support and requirements elicitation.