Best Practices to Achieve CMMI Level 2 Configuration Management Process Area through VSS tool

Prerna Gupta
prernagupta2k7@gmail.com
Co-Author
Dr.D.S.RAO
Dr.dsrao@yahoo.in

Abstract

Over the past years, the capability maturity model (CMM) and capability maturity model integration (CMMI) have been broadly used for assessing organizational maturity and process capability throughout the world. However, the rapid pace of change in information technology has caused increasing frustration to the heavyweight plans, specifications, and other documentation imposed by contractual inertia and maturity model compliance criteria. In light of that, configuration management methodologies have been adopted to tackle this challenge. The aim of our paper is to present mapping between CMMI and one of the SCM tool VSS. It shows how VSS addresses the Configuration Management Process Areas of CMMI and help in achieving all the goals which are defined at CMMI level 2 Configuration process areas. This is useful for organizations that have their plan-driven process based on the CMMI model and are planning to improve its processes toward agility or to help organizations to define a new project management framework based on both CMMI and configuration practices.

1. Introduction

Process improvement is a series of actions taken to identify, analyze and improve existing processes within an organization to meet new goals and objective. These actions follow a specific methodology or strategy.

When developing software, you need to have a manageable team development effort, track and maintain the history of your projects, sustain parallel development on multiple product versions, fix bugs, and release service packs while further developing the application. This is where the concepts of Software Configuration Management come into play.

Software Configuration Management (SCM) is the collection of processes and tools that are used to effectively manage the development, maintenance, and build processes of software. Tools alone will never get the job done. It is important to note that processes or "best practices" are a critical element in the total success of SCM for your organization.
Using good software configuration management tools will help you ensure that you maintain all the artifacts for the project in one location.

2. Objective

The aim of our paper is to present mapping between CMMI Configuration management Process area and one of the SCM tool VSS. This CM paper will describe how a company, organization, or project can leverage the Processes and tools provided by Microsoft for achieving CMMI ML 2. At a high level, it describes how the CMMI views Configuration Management, and discusses how VSS tool solutions satisfy the Configuration Management Process Area’s specific and generic goals and practices. The scope of this paper is the Capability Maturity Model Integration (CMMI) Configuration Management process area maturity level 2 (CMMI ML 2) Requirements. In cases where Microsoft VSS solutions do not fully address a practice, recommendations are made on how the organization might satisfy the practices to attain CMMI ML 2 compliance. At the end of this paper a matrix is provided listing each CMMI CM goal/practice, its description, the VSS tool role that might be responsible for satisfying the practice, what VSS process, procedure, template, or VSS tool might help satisfy the practice and any other observations and recommendations that might be relevant.

3. Assumptions

The CMMI appraisal model is a verification-based audit that essentially follows the “Say what you do, do what you say, prove it” concept. This means that organizations that wish to achieve a CMMI maturity level 2 rating must document various management and engineering practices, execute them as documented, and be able to prove it by the existence of quality records and/or project artifacts. During an appraisal, an SEI Authorized Lead Appraiser and qualified appraisal team members look for objective evidence that the organization has implemented and institutionalized the practices. Objective evidence consists of direct artifacts, indirect artifacts, and affirmations. Direct artifacts are tangible work products directly created as a result of implementing a practice (e.g. a project plan or CM plan). Indirect artifacts are a side effect of implementing the practice or otherwise indicate the practice was performed (e.g., meeting minutes, reviews, logs). Affirmations are oral or written statements confirming or supporting the practice was implemented.

This paper makes the following organizational assumptions to show how VSS solutions are utilized:

1. Adoption and implementation of the VSS
2. Use of the VSS as documented
3. Creation of the artifacts as defined by the VSS

This paper will identify the various VSS roles, disciplines, templates and activities that would apply in satisfying the various CMMI maturity level 2 CM key practices. The process area (PA) matrix attached to this paper lists each CMMI CM practice by goal (both specific and generic), its description, the VSS role that might be responsible for satisfying the key practice, what VSS process and procedure/template might help satisfy the key practice, and any notes that might be relevant. In situations where VSS does not fully address a key practice; suggestions are made in the notes column of the matrices on how the project might address the key practice to attain compliance.

4. Description of CMMI Configuration Management Process Area’s Specific Goals
The purpose of Configuration Management (CM) is to establish and maintain the integrity of work products using configuration identification, configuration control, configuration status accounting, and configuration audits.

The Configuration Management process area involves the following specific goals:

- Identifying the configuration of selected work products that compose the baselines at given points in time
- Controlling changes to configuration items
- Building or providing specifications to build work products from the configuration management system
- Maintaining the integrity of baselines

SG 1 Establish Baseline:
Baseline is an object that typically represents a stable configuration of a component. A baseline is a set of specifications or work products that has been formally reviewed and agreed on, that thereafter serves as the basis for further development or delivery, and that can be changed only through change control procedure. A baseline identifies activities and one version of every element in a component, in effect, acting as a version of a component.

For System Engineering, Baseline can be viewed as: The system-level requirements, system-element-level design requirements, and the product definition at the end of development/beginning of production. These are typically referred to as the “functional baseline,” “allocated baseline,” and “product baseline.”

For Software Engineering, Baseline can be viewed as: A Software Baseline can be a
set of requirements, design, source code files and the associated executable code, build files, and user documentation (associated entities) that have been assigned a unique identifier.

Below are the Practices which can be followed to achieve this goal:

1.1 Identify Configuration Items

Configuration identification is the selection, creation, and specification of the following:

- Products that are delivered to the customer.
- Designated internal work products.
- Acquired products.
- Tools and other capital assets of the project's work environment.

Other items that are used in creating and describing these work products.

Implementation Approach:

When any member of the project team creates any new item of project documentation that requires distribution to a variety of other project staff, they should submit it to the configuration administrator, or whoever is designated as being responsible for administering the configuration management procedures - in order that they can decide whether or not it constitutes a configuration item (CI). Once the item is designated as the configuration item-it can be uploaded through VSS tool in its database. A configuration item may have a number of versions relating to the continuing development of that configuration item. Initially the first version of the approved baseline will be uploaded by the configuration manager. The configuration item may then undergo a number of modifications as it is tested and errors corrected, or as changes are requested. As changes require an initial state and next state, the marking of significant states within a series of several changes becomes important. So, with the help of VSS the identification of significant states within the revision history of a configuration item can be done which is the central purpose of baseline identification.

History of a configuration item identifies the versions of that particular item. A new version will be created whenever a change is made to an existing item.

Sub Practice 1.1.1: Select the configuration items and the work products that compose them based on documented criteria.

Implementation Approach:

To relate with the real IT world, here is an example-

In an organization where there exists a testing project we maintain the below artifacts in the mentioned sub-folders within the main project folder:
Deliverables: In this folder, we put all those documents which are sent to the client after the test execution.
   1) Test Plan and Test set
   2) Test Summary
   3) Requirement Traceability Matrix
   4) Lesson Learnt

Receivables: In this folder, we put all those documents which are received from the client before the test execution.
1) Base lined BR, SRS, HLD, wireframe docs.

2) Scope Change Documents

VSS has the functionality of retrieving the documents basis on their LABELS which also helps in identifying the baseline.

Naming Convention: Project number _Team Name_Document name

Below are the following documents which we used within an organization for a testing project.

RTM-184113a_DMDR Dashboard_RTM
Test Plan-184113a_DMDR Dashboard_Test Plan
Data Request-184113a_DMDR Dashboard_Data Request

The following screenshot represents how we can create the different folders in VSS so that documents can be maintained, stored and accessed whenever required.
Sub Practice 1.1.2: Assign unique identifiers to configuration items.

Implementation Approach
The entity must be uniquely identified so that it can be distinguished from all other configuration items i.e. unique identifiers are assigned to the configuration items so that we can differentiate among the different versions of the same documents as well as to differentiate among the different documents.
The following naming convention can be used to identify the different documents which will act as the label for them. **VSS has the functionality of retrieving the documents basis on their LABELS.**

**Naming Convention: Project number _Team Name_Document name**

Below are the following documents which we used within an organization for a testing project.

RTM-184113a _DMDR Dashboard_RTM

**Sub Practice 1.1.3:** Specify the important characteristics of each configuration item.

**Implementation Approach:**

Through the VSS we can specify the important attributes of a configuration item like who checked out or checked in the document and the date which tells about when the user check out and check in again. The new and the old versions of a document are available for the user.
Sub Practice 1.1.4: Specify when each configuration item is placed under configuration management.

Implementation Approach:

With the help of VSS we can label the documents in such a way that can represent the stage in which we create the document.

The following convention can be used:

The identifier consists of three parts separated by hyphens in the format 123-45-67:

The first part of the identifier is the project number; the second part identifies the stage in which the item was originally created; the third part is simply an ascending order for items generated during the stage.

217-02-02

Project Stage Sequential Number

Stage numbers can be assigned as follows:

Configuration Items:

00 Reference items, such as the SDLC and glossary
01 Planning stage items
02 Requirement Stage Items
03 Design stage Items
04 Development stage Items
05 Test and Acceptance stage Items

These stage numbers are already communicated through mail through the whole team working on the project. Either we can directly use the description of the stage in which the document is uploaded on the VSS.

For example: The requirement traceability matrix is first created during the requirement stage and is generally created second after the requirement documents. For project 217 the requirement traceability would carry an identifier of 217-02-02 or 217-Planning-02. This is known as item’s identifier.

Sub Practice 1.1.5: Identify the owner responsible for each configuration item.

Implementation Approach

Through VSS we can represent the Owner who checks in the document or who has made the changes in the document.

SP 1.2 Establish a Configuration Management System

Sub Practice 1.2.1: Establish a mechanism to manage multiple control levels of configuration management.

The level of control is typically selected based on project objectives, risk, and/or resources. Control levels may vary in relation to the project lifecycle, type of system under development, and specific project requirements.

Sub Practice 1.2.2: Store and retrieve configuration items in a configuration management system.

Implementation Approach:

Through VSS, we can retrieve the configuration item through Check out Command. To make changes to a file you must first check it out of the VSS database. When you Check Out an item, VSS retrieve the latest copy of the file to your working folder and make it writable.

There are exclusive check out and non-exclusive check out. If a file is checked out exclusively, it cannot be checked out by other users.
A working folder is the specified corresponding folders on a user’s local computer used to store files when working with VSS projects. By default, the files are retrieved to the corresponding working folder when a user does get or Check Out. A working folder is based on per project, per user and per machine. A user can make changes to files in the working folder and then check the modified files back into the VSS database for version tracking. Check-in Command enables the user to store the configuration item in the VSS database. After you have finished the changes, you need to Check In the updated file back into VSS. When we say VSS keeps all your previous changes, we are not saying that VSS keeps every change you make, we are saying that VSS keeps all the versions you checked in. If you do not check in your file, there is no way that VSS knows that you have changed the file and it should keep it.

**Sub Practice 1.2.3:** Share and transfer configuration items between control levels within the configuration management system.

**Implementation Approach:**

In VSS, Share command enables files to be shared among multiple projects. It creates share links among these projects, so that the item can be viewed in all projects. If an item is modified in one project, the changes will be reflected in other projects simultaneously.

**Sub Practice 1.2.4:** Store and recover archived versions of configuration items.

**Implementation Approach:**

Visual SourceSafe provides an Archive utility, with which we can periodically backup our VSS database or projects and transport files/projects between SourceSafe databases. SourceSafe also provides a Restore tool which allows us to restore the data from an archive.

**Sub Practice 1.2.5:** Store, update, and retrieve configuration management records.

**Implementation Approach:**

A history is maintained whenever we check out and check in the configuration items. In history we can determine who retrieve the configuration item and when? What modifications has ben performed? So in ths way we manage the configuration management records about the configuration tem.

**Sub Practice 1.2.6:** Preserve the contents of the configuration management system.

**Examples of preservation functions** of the configuration management system include the following:

- Backups and restoration of configuration management files.
- Archiving of configuration management files.
- Recovery from configuration management errors.
Implementation Approach:

As already mentioned above we have determined how to use the archive command in VSS which will be useful for providing the backups of the project.

SP 1.3 Create or Release Baselines
For this particular goal to be achieved there is no role of VSS particularly rather it’s the duty of Configuration manager and the team to ensure that the latest documents are uploaded in VSS which can be shared among the team members working on that same project so that no confusion would creep up among the team members and all of will do the work on the same documents.

Sub practice 1.3.1: Obtain authorization from the configuration control board (CCB) before creating or releasing baselines of configuration items.

Implementation Approach:

When any member of the project team creates any new item of project documentation that requires distribution to a variety of other project staff, they should submit it to the configuration administrator, or whoever is designated as being responsible for administering the configuration management procedures - in order that they can decide whether or not it constitutes a configuration item (CI). Once the item is designated as the configuration item-it can be uploaded through VSS tool in its database. A configuration item may have a number of versions relating to the continuing development of that configuration item. Configuration manager must ensure that whenever any document is uploaded in the VSS, it must be approved by the authorized approvers or the client. The configuration item may then undergo a number of modifications as it is tested and errors corrected, or as changes are requested.

Sub Practice 1.3.2: Create or release baselines only from configuration items in the configuration management system.

Implementation Approach:

Lets suppose there is an IT organization with the testing project then the team can ensure while testing all the requirements it should have all the latest versions of configuration item uploaded in VSS database .All the documents like SRS (Software Requirement Specification), BR(Business Requirements) received from the client are uploaded in VSS.

Sub Practice 1.3.3: Document the set of configuration items that are contained in a baseline.
Implementation Approach:

Before starting the testing any project we should ensure that all the latest documents (BR, SRS, SA, HLD) are received from the client and uploaded in VSS.

Sub Practice 1.3.4: Make the current set of baselines readily available.

Implementation Approach:

When we received the documents from the client we can check in those documents in the VSS database. Once the documents are uploaded in the VSS database, they are accessible to all the team members working on the project by providing them the appropriate access.

SG 2 Track and Control Changes

In this goal we keep the track of change request from their initiation to their closure.

A change request is a proposal from a stakeholder in the software development process to change something in a product or in a product process. Common change requests include requests for product enhancements or new features. A change request can be referred as a scope change. Change requests are analyzed to determine the impact that the change will have on the work product, related work products, budget, and schedule. In practice, the basic details of a change request from the business are recorded to initiate the change process, including references to documents that describe and authorize the change.

Implementation Approach: VSS is a central repository which can be used by the change manager as well. They can update and maintain their records in the database. Whenever there is a request of a scope change, the RFC (request change form) can be filled by the requester. RFC is a form which is used to maintain the scope changes of all the documents. Change manager can upload the RFC form in a folder named as a Scope change in the VSS database. Change manager and the other team members working on the project have access to that folder. However the write access is given only to Change Manager for that particular folder.

The Label can be used while uploading the RFC document which will describe about the status of scope change. Initially the status can be put as Change Request. The business owner of a configuration item (i.e., a CI in the Configuration Management Database which records the exact state of the IT infrastructure) to be changed (e.g., IT for infrastructure, Finance for a Billing application, etc.) and all affected groups (e.g., users,
management, IT, etc.) are identified and asked to contribute to an assessment of the risk and impact of a requested change.

Once the Change manager get the analyze impact from the business owner they can update the status of RFC form in the database. Now the label which is used for describing the status of the RFC form can be put as “PENDING”.

A change should normally be made by a change owner within the group responsible for the components being changed. A release or implementation plan should be provided for all but the simplest of changes and it should document how to back-out or reverse the change should it fail. On completion of the change the results should be reported back for assessment to those responsible for managing changes, and then presented as a completed change for customer agreement. Once the change manager gets the approval from the customer for the scope change, the label can be changed from Pending to Approved state. If the scope change is not approved by the customer, the status can be changed to “Not approved” state.

The change request and configuration management database should be updated, so that the person who initiated the change is aware of its status. Actual resources used and the costs incurred are recorded as part of the record. A post-implementation review should be done to check that the completed change has met its objectives, that customers are happy with the results; and that there have been no unexpected side-effects. Lessons learned are fed back into future changes as an element of continuous process improvement.

Once the approval has been obtained, the Change requester can close the request and changed the status of request as to a “Closed state”.

The person who initiated the change request should consistently check the status of the RFC document. Once the status of the Request has reached to the closed state the document can be provided to the whole team and upload in the Project’s folder. From where it is accessible to every team member.

Control Configuration Items

Control is maintained over the configuration of the work product baseline. This control includes tracking the configuration of each of the configuration items, approving a new configuration if necessary, and updating the baseline. Change management should also work closely with configuration management to ensure that all changes to IT components are properly documented in the configuration management database, which is a repository used to track the status of all components of the IT environment. Configuration manager ensures that the latest version of the document has been uploaded in the VSS. The configuration management database acts as a historical record of changes to the environment and maintains a "picture" of the existing IT environment. All the versions of the document can be found by checking their history in the VSS.
The following screenshot represents how the history is maintained in order to track the changes made in the document.

### SG 3 Establish Integrity

Integrity means ensuring the correctness of the document.

#### Implementation Approach

VSS helps in establishing and maintaining records by describing configuration items. VSS records configuration management actions in sufficient detail so that the content and status of each configuration item is known and previous versions can be recovered.

Within the VSS, we have the option to provide access permissions to authorized end users. VSS saves the document in an encrypted form. So in this way we can maintain the integrity of the documents. As there is the provision of username and password only authorized users can modify the documents. We provide different levels of control level access to different users.

Configuration audits consist of cross-checking the CI description records with the current version of the CI as held in its development folder. These audits are normally carried out ahead of each project review, although they can be requested at any time by the project manager.
manager or the project owner. Configuration audits should establish that: the current physical representation of each CI matches its current specification, that there is consistency in design between each CI and its parent, if there is a parent. Finally it also establishes that the documentation is up to date and all relevant standards are being adhered to. The configuration administrator should be responsible for producing all scheduled and ad hoc reports on the status of the CIs at any given point in time. The status of the document can be viewed in VSS.

The following screenshot represents how we can provide the different levels of control level access to different users.

5. Conclusion

By using all the Best practices specified above in the paper, VSS can achieve the CMMI level 2 Configuration management Goals in a very cost effective manner. There are many SCM tools on the market today and their capabilities and prices range significantly. **While choosing a SCM tool make ensure what are your requirements?**

- Is it laid out in your company’s process?
- Does the customer require a certain product?
• Reliability and support of the product
• Appearance
• (GUI front end)
• Integration with specific development environment
• Platform (operating system)
• Cost
• Other functionality version control.

**VSS, is the superior choice because, it provides**
• Flexibility within a single product family, one that offers ease-of-use and powerful features for more advanced teams as well as small teams.
• Easy to set up, easy to use, integrates with Visual Studio.
• A company can achieve the CMMI level 2 configuration Process area if used VSS.

By using VSS organization can deliver a more bug fee product, automate repetitive development tasks create a more predictable software development and release cycle manage concurrent development of multiple source files /projects which contributes to increased productivity create the distributed development teams regardless of the location.

**References**


• Jonathan B.Steele, Software Configuration Management tools Dec 7, 2002 University of Maryland University, College MSIT 630, section 1131

• Dennis R. Golden son, Demonstrating the Impact and Benefits of CMMI October, 2003

• Introducing CMMI for Development Produced by Det Norske Veritas October, 2009


• Herve Rochecouste, CMMI-Use the Body of Knowledge to Create and Improve your System Integration Capability, December 2001


