Configuration Management Process

SOC 2 Document

esm_logo.jpg

Nexelus USA  
New York, NEW YORK (NY)  
646-558-1950 ext.128

**10/26/2022**

# Document Information

The following table shows the details for document creation, review, approval, and effective date.

| **Category** | **Information** |
| --- | --- |
| Work Product: | Configuration Management Process – SOC 2 Document |
| Project Name: | Configuration Management Process |
| Function Name: | SOC 2 Document |
| Version: | 1.0 |
| Status: | Draft |
| Author(s): | Tauseef Shahzad |
| Reviewer(s): | Asim Jameel |
| Approver(s): | Imran Rahman |
| Control Status: | CONTROLLED, PROTECTED |
| Disclaimer: | This document contains confidential information. Do not distribute this document without prior approval from Nexelus. |

# Revision History

The following table is used for revision details of this document.

| **Author(s)** | **Date** | **Version** | **Description of Change** |
| --- | --- | --- | --- |
| Tauseef Shahzad | October 10, 2021 | 1\_0 | Initial Draft |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Table of Contents

Contents

[Document Information ii](#_Toc117762832)

[Revision History iii](#_Toc117762833)

[Table of Contents i](#_Toc117762834)

[Scope 1](#_Toc117762835)

[Configuration Management 2](#_Toc117762836)

[Software Configuration Management 3](#_Toc117762837)

[Configuration management in a development cycle 3](#_Toc117762838)

[Release Backlog and Issue Tracking 4](#_Toc117762839)

[Tasks in development and maintenance 4](#_Toc117762840)

[Configuration Identification 6](#_Toc117762841)

[Identification rules of configuration items 6](#_Toc117762842)

[Identification rules of documents 6](#_Toc117762843)

# Scope

Software Configuration Management(SCM) is a process to systematically manage, organize, and control the changes in the documents, codes, and other entities during the Software Development Life Cycle. The primary goal is to increase productivity with minimal mistakes. SCM is part of cross-disciplinary field of configuration management and it can accurately determine who made which revision.

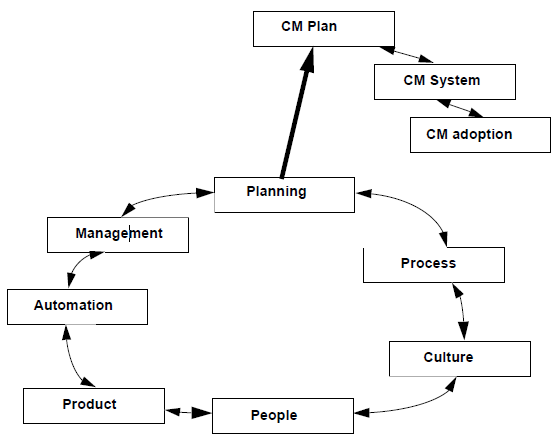
The scope of this document is to document Software Configuration Management Process as implemented for Nexelus Software Development.

**Reference**

SSAE-18 SOC 1 Type II – Requirements

# Configuration Management

There are ten elements which are the keys to solving the CM needs in an organization. Seven of these elements relate to the problem preparation and solving work and the other three are the results of seven element. All 10 elements are shown in Figure below:



## Software Configuration Management

At Nexelus, the software configuration management is done with Team Foundation Server, a SCM tool that integrates with Microsoft Visual Studio as well as provides a web interface/ Team Foundation Server provides Source Code Management as well as acts as primary bug tracking and task management tool.

For each project a root branch is created with template “\_<ProjectID>Trunk” as main development branch, called the Master, receives by default all software developments made by the software team. When a new major version is planned (for instance V1.0 or V2.0), a branch is created from the master. This branch is isolated to be tested, fixed, and finally delivered. A Minor branch is assigned an incremental decimal value such as (1.1, 1.2, …).

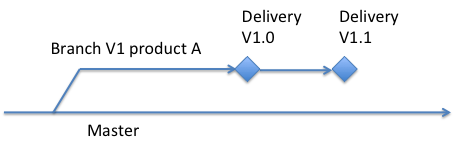


Figure Master and branches in Team Foundation Server

### Configuration management in a development cycle

The changes made by developers during a development cycle are managed by the following method.

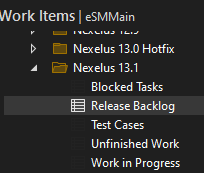
Once the work on a branch is complete it is marked as “Read Only” and merged into Master branch.

Two new branches are created from Master branch:

1. New branch with new release version. All new development is maintained in new branch.
2. Hot Fix Branch with existing release ID and Hotfix prefix. All production issues related to current deployed release are checked into this branch.

Periodically the Hotfix branch is merged into Master branch and then from Master branch, merged into new Development Branch.

### Release Backlog and Issue Tracking



A new Iteration is created in TFS “Work Items” corresponding to Development release.

* All Backlog items are listed in this iteration status “New”.
* Once the Backlog item is assigned to a developer, it is marked as “approved”
* The developer marks the backlog item as Committed, after the domain Transfer session.
* The QA team starts entering test cases against Backlog items in parallel.
* When Developer starts development, he marks it as “In Progress”
* Once the Development is complete, the developer marks the Backlog item as “Review” for unit testing and review by Team Lead.
* After reviewing the backlog item, it is marked as Testing and assigned to QA Team.
* The QA Team tests the backlog item. If no issues are found, it is marked as “Demo”
* If some issues are found, the status is changed to “Bug Fixing” and assigned back to developer.

### Tasks in development and maintenance

The tasks depend on the phase of the software development project or of maintenance. The SCM Manager does the following operations, in the software life-cycle.

|  |  |  |
| --- | --- | --- |
| **Item** | **Event** | **Operation** |
| 1 | Launching the development of a new product | Creating the source folder structure in the master branch |
| 2 | Deciding to create a major version | Fork of a branch from the current state of the master branch |
| 3 | Releasing a major version | Tagging the current version in its branch.  Archiving the tagged version |
| 4 | Releasing a minor version or a patch | Adding a new tag to the current version in the branch.  Archiving the tagged version |
| 5 | Closing an iteration cycle | Adding a new tag to the current version in the master branch |

## Configuration Identification

### Identification rules of configuration items

The identification of configuration item is:

* <configuration item name>\_m.n

where:

* "m " is the major version of the configuration item,
* n is the minor version number,

The version number of the configuration item m.n starts at 1.0.

The number "m" of major version is incremented when substantial modifications are made to the device, for example:

* Updating of the intended use,
* Adding new modules / functionalities,

The number "n" of minor version is incremented when non-substantial modifications are made to the device, for example:

* Adding new functionalities to existing modules,
* Updating the GUI.

### Identification rules of documents

The identification of documents is described below:

XXX-<document number>\_<revision index>

where:

* XXX is an acronym to identify the project/module/functionality
* " document number " is a incremental in the project,
* " revision index " designates the approved iteration of the document. The revision index is 0 for draft version, 1 for the first approved revision, 2 for the second and so on.