

**Campus Meal Ordering System**

**Software Configuration Management Plan**

**By *Team Foodie***

**Lab Group: TS3**

**Date: October 2020**

**Instructors:**

**Bo An**

**Leong Yin Yoke Junie**

**Qiu Wei**

**Wang Rundong**

# SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

**NANYANG TECHNOLOGICAL UNIVERSITY**

**TABLE OF CONTENTS**

1 Introduction 2

1.1 Document overview 2

1.2 Abbreviations and Glossary 2

1.2.1 Abbreviations 2

1.3 References 2

1.3.1 Project References 2

1.3.2 Standard and regulatory References 2

2 Organization 2

2.1 Activities and responsibilities 3

2.1.1 Decisions process and Accountabilities 3

3 Configuration identification 4

3.1 Identification rules 4

3.1.1 Identification rules of configuration items 4

3.1.2 Identification rules of SOUPs 4

3.1.3 Identification rules of documents 4

3.1.4 Identification rules of a media 5

3.2 Reference configuration identification 5

3.3 Configuration Baseline Management 5

4 Configuration control 5

4.1 Change Management 5

4.2 Interface Management 6

4.3 Evolutions control of SOUP items 6

5 Configuration support activities 6

5.1 Configuration Status Accounting 6

5.1.1 Evolutions traceability 6

5.1.2 Setting up Configuration status 6

5.1.3 Configuration status diffusion 6

5.1.4 Configuration status records storage 6

5.2 Configuration audits 7

5.3 Reviews 7

5.4 Configuration management plan maintenance 7

# 

# 

# 

# 

# Introduction

## Document overview

This document contains the software configuration management plan of the Campus Meal Ordering System (CMOS).

## Abbreviations and Glossary

### Abbreviations

CMOS: Campus Meal Ordering System

## References

### Project References

| # | Document Identifier | Document Title |
| --- | --- | --- |
| R1 | D01 | Use Case Description |
| R2 | D02 | System Requirement Specification |
| R3 | D03 | Quality Plan |
| R4 | D04 | Project Plan |
| R5 | D05 | Risk Management |
| R6 | D06 | Change Management Plan |
| R7 | D07 | Release Plan |
| R8 | D08 | Design Report for Software Maintainability |
| R9 | D09 | Software Configuration Management Plan |

### Standard and regulatory References

| # | Document Identifier | Document Title |
| --- | --- | --- |
| STD1 | D10 | CMMI Processes Definition |
| STD2 | D11 | IEEE 829 Test Plan |

# 2. Organization

The software configuration is managed by members of the project, with specific tools. Responsibilities are shared between

* The software configuration manager (SCM),
* The project manager,
* The release engineer

## 2.1 Activities and responsibilities

Describe here the functions required to manage the configuration of the software and responsibilities.

| **Activities when setting up the project** | **Person responsible** |
| --- | --- |
| Identify the configuration items | Release Engineer |
| Install the bug repository tool and set up the database | Backend Developer |
| Install the software configuration repository tool and set up the database | Release Engineer |
| Manage and structure the reference space | Release Engineer |
| Define the configuration processes | Release Engineer |

| **Activities during the project lifecycle** | **Person responsible** |
| --- | --- |
| Export components for modification, test or delivery | Release Engineer |
| Set under control validated components | Release Engineer |
| Create version, write version delivery document | Release Engineer |
| Approve reference configurations | Project manager |
| Verify version to be delivered and authorise deliveries | Project manager |
| Backup spaces | Release Engineer |
| Do configuration audits | Quality Manager |
| Inspect configuration records | Quality Manager |
| Archive reference version | Release Engineer |

| **Management activities** | **Person responsible** |
| --- | --- |
| Manage versions and archives | Release Engineer |
| Manage configuration records | Release Engineer |
| Produce reports and statistics | Release Engineer |
| Manage reference space and its access control list | Release Engineer |
| Manage spaces backup and archive media | Release Engineer |
| Manage quality reports | Quality Manager |

### 2.2 Decisions process and responsibilities

Responsibilities during reviews, audits and approvals are listed below:

At the end of an activity of the project

| **Activities** | **Person Responsible** |
| --- | --- |
| Do a configuration freeze | Project manager |
| Present a configuration state of the components impacted by the activity | Release Engineer |
| Present a documentation state of the components impacted by the activity | Release Engineer |

During a configuration management process audit:

| **Activities** | **Person Responsible** |
| --- | --- |
| Do the configuration management process audit | Project Manager |
| Present the records of the configuration management process | Release Engineer |
| Present the quality records of the configuration management process | Quality Manager |
| Present the records of the documentation management process | Release Engineer |

# 3. **Configuration identification**

## 3.1 Identification rules

Below are the identification rules which must be adhered to minimize confusion during the development of the project.

### 3.1.1 Identification rules of configuration items

#### Identification of a configuration item

The identification of configuration item is as follows:

* <project\_name>\_<iteration>\_<date> where date is formatted as MM\_DD\_YY.

#### Version number of a configuration item

The attribution of a version number is a prerequisite to any delivery of any configuration item. This number shall be incremented before a new delivery, if the product or its documentation were modified.

* Version numbering will follow Semantic versioning.
* The sequence number will use 3 separate numbers separated by full stops denoting 3 types of software iterations
* Major.Minor.Patch eg. 1.8.1

### 3.1.2 Identification rules of SOUPs

#### Identification of a SOUP

IDs will be based on manufacturer’s identification

#### Version number of a SOUP

SOUP version numbers will be based on manufacturer’s version naming conventions

### 3.1.3 Identification rules of documents

#### Description of documents identifiers

The identification of documents is described below:

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

where:

" document type " is:

* PP, for project plan documents
* RM for risk management documents
* QA, for quality plan documents
* SR, for system requirement documents

" document number " is a incremental number, with a separate list for each document type,

" revision index " designates the approved iteration of the document. The revision index is V1 for the first iteration, V2 for the second and so on.

#### Definition and evolution of the revision index

The attribution of a revision index is a prerequisite to any delivery of a document or file. This index shall be incremented before the diffusion of a modified document.

The definition rules of a revision index are the following:

* Patch

Changes in formatting or exporting

* Minor revision

Typing mistakes, spelling mistakes, translation mistakes or addition of information where the added information does not change the meaning of the document.

* Major revision

All changes that do not fall under the scope of a patch or minor revision including adding, rectifying and overhauling information or documents

### 3.1.4 Identification rules of a media

#### Internal identification

The identification of a media is described below:

<configuration item identification >/<media>/<volume>

where:

"media" is the media number,

"volume" is an incremental number to distinguish the media if the delivery contains more than one media.

## Reference configuration identification

Each reference configuration is defined by:

* An identifier,
* Its content listed in the corresponding Version Delivery Description document,
* The acceptance or validation reviews associated with the building of the reference configuration.

A reference configuration is established for each design review and each test review of the project.

## 3.2 Configuration Baseline Management

Describe what baselines are to be established. Explain when and how they will be defined and controlled.

Examples of baselines :

* functional baseline (FBL), which describes the system functional characteristics;
* allocated baseline (ABL), which describes the design of the functional and interface characteristics,
* product baseline (PBL), which consists of completed and accepted system components and documentation that identifies these products.

# 4. Configuration Control

Configuration changes and variances in configurations will be proposed by any team member and will be vetted by the release engineer and project manager to be integrated.

## 4.1 Change Management

Minor and major changes to the baseline will be archived for no less than one year and will be tracked independently by both the project manager and release engineer in the form of a google doc.

Example for problem resolution:

* Changes requests are emitted from by the project manager according to the problem resolution process,
* When a change request is accepted by the project manager/product manager, a branch is created in the SCM
* The branch identification is dependent on the branch name
* Branch content is reviewed by the team and either merged or rebased

Example for multiple configuration:

* Changes requests of configuration files are emitted by the product manager according to the production procedure
* When a change request is accepted by the project manager/release engineer, a branch is created in the SCM
* The branch identification is dependent on the purpose for the branch
* Branch content will be dependent on baseline and additional changes

## 4.2 Interface Management

List of interfaces:

* Google drive (Google)
* Firebase (Google)

## 4.3 Evolutions control of SOUP items

SOUPS will be frozen at the start of development and will be updated as necessary dependent on the continuity of the project’s functionality, risk of threats, SOUP version release cycle and requirements of the product

# 5. Configuration Support Activities

## 5.1 Configuration Status Accounting

Configuration Status Accounting (CSA) is the process to record, store, maintain and report the status of configuration items during the software lifecycle. All software and related documentation should be tracked throughout the software life.

### 5.1.1 Evolutions traceability

The traceability of modifications of items given their types:

* Document: The modification sheet number identifies the origin of the modification. The modified paragraphs in the document are identified, if possible, by revision marks.
* Source file: The software configuration management tool records, for each source file or group of source files, a comment where the modification is described.
* Configuration item: The Version Delivery Description of the article identifies the modification sheet included in the current version.

The modification sheet describes the modifications done to the components with enough precision to identify the modified parts.

### 5.1.2 Setting up Configuration status

The SCM sets up the state of all versions and of each configuration article with:

* The label,
* The version number,
* The creation date of the VDD,

The SCM writes the VDD.

### 5.1.3 Configuration status diffusion

The SCM and the quality manager write the VDD.

### 5.1.4 Configuration status records storage

The records are stored in a configuration folder, which contains:

* The requests sorted by record number,
* The software documents,
* The VDD’s,
* The configuration states sorted chronologically.

## 5.2 Configuration audits

Peer review audits will be handled by both project manager and release engineer and formal audits will be managed by the project manager

List of audits:

* Baseline audit.
* Functional configuration audio
* Software configuration audit
* Backup and risk management audit

## 5.3 Reviews

Each branch will conduct periodic technical reviews on their assigned branch and will be vetted by the project manager and release engineer. Baselines will be updated and refreshed by the release engineer under the purview of the project manager

## 5.4 Configuration management plan maintenance

Update frequency of branches will be handled by the persons assigned to specific branches to be approved by both release engineer and project manager. Baselines will have a fixed update schedule to be determined by the project manager and responsibility will fall on the release engineer to be implemented