

USEFUL JAPANESE DICTIONARY FOR VIETNAMESE

CM Plan

**Project Code: UJD\_VN**

**Document Code: UJD\_VN\_CMPLan\_v1.0\_EN**

**Ha Noi, 06/06/2014**

SIGNATURE PAGE

AUTHOR: Nguyễn Ngọc Tuấn 06/06/2014

Team member

REVIEWERS: Nguyễn Ngọc Tuấn 06/06/2014

Team member

APPROVAL: Nguyễn Văn Sang 09/06/2014

Supervisor

Record of change

\*A - Added M - Modified D – Deleted

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Effective Date | Changed Item | A\* M, D | Change Description | Reason for Change | Revision Number |
| A | Add new | A |  |  | v1.0 |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

TABLE OF CONTENTS

[1. Introduction 5](#_Toc251245974)

[1.1. Role & Responsibility 5](#_Toc251245975)

[1.2. Definitions and Acronyms 5](#_Toc251245976)

[2. configuration management Process 6](#_Toc251245977)

[2.1. CI Identification & Naming convention 6](#_Toc251245978)

[2.2. Project Infrastructure 6](#_Toc251245979)

[2.3. CI Baseline Procedure 7](#_Toc251245980)

[2.4. Directory structure & Access right 8](#_Toc251245982)

[2.5. Version numbering rule 10](#_Toc251245983)

[2.6. Other CM rules 11](#_Toc251246020)

# Introduction

The purpose of this document is to identify and describe Configuration management (CM) process implementing in the project.

## Role & Responsibility

Refer to Project Organization section in [Project Plan](JSBC_ProjectPlan_v0.1_EN.docx)

## Definitions and Acronyms

| Acronym | Definition | Note |
| --- | --- | --- |
| UJD\_VN | Useful Japanese Dictionary for Vietnamese |  |
| ADD | Architecture Design Document |  |
| CI | Configuration Item |  |
| CM | Configuration Management |  |
| PP | Project Plan |  |
| CSCI | Computer Software Configuration Items |  |
| DDD | Detail Design Document |  |
| PM | Project Manager |  |
| QA | Quality Assurance Officer |  |
| SRS | Software Requirement Specification |  |
| Source | Source Code |  |
| URD | User Requirement Document |  |
| TP | Test Plan |  |
| PIC | Person in Charge |  |

# configuration management Process

## CI Identification & Naming convention

| CSCI | Configuration Items | Naming Conventions |
| --- | --- | --- |
| Project Management | PP | UJD\_VN\_ProjectPlan\_v<x.x> \_language |
| CM Plan | UJD\_VN\_CMPlan\_v<x.x>\_language |
| Introduction (Report 1) | UJD\_VN\_Introduction\_EN |
| Risk Management | UJD\_VN\_Risk Management\_language |
| Task List | UJD\_VN\_Tasklist |
| Project Schedule | UJD\_VN\_Project\_Schedule\_v<x.x> \_language |
| Examination | UJD\_VN\_Examination\_v<x.x>\_language |
| Requirement & Design | Software Requirement Specification | UJD\_VN\_Software requirement specification\_v<x.x> \_language |
| Screen Design | UJD\_VN\_Screen design\_v<x.x>\_language |
| Architecture Design | UJD\_VN\_Architecture Design\_v<x.x>\_language |
| Class Design | UJD\_VN\_Class Design\_v<x.x>\_language |
| Data Design | UJD\_VN\_Data Design\_v<x.x>\_language |
| Prototype | Prototype\_v<x.x> |
| Build | Source code | UJD\_VN\_Source Code\_v<x.x> \_Tested/Untested |
| Test Plan | UJD\_VN\_Test Plan\_v<x.x>\_language |
| System Test Case | UJD\_VN\_System Test Case\_v<x.x>\_language |
| Integration Test Case | UJD\_VN\_Integration Test\_v<x.x>\_language |
| Unit Test Case | UJD\_VN\_Unit Test Case\_v<x.x> \_language |
| Defect Log Management | UJD\_VN\_Defect Log Management\_v<x.x>\_language |
| Component Test Case | UJD\_VN\_Component Test Case\_v<x.x> \_language |
| Test report | UJD\_VN\_Test Report\_v<x.x>\_language |
| Process | Guideline\_Name of guideline | UJD\_VN\_Guideline\_Title of guideline |
| Coding Convention/Stadard | UJD\_VN\_Coding convention |
| Checklist\_Name of checklist | Checklist\_Title of checklist\_v<x.x> |
| Document type | MS Word | \*\*\*.docx |
| MS Project | \*\*\*.mpp |
| MS Excel | \*\*\*.xls |

## Project Infrastructure

| Tool | Version | Purpose | Note |
| --- | --- | --- | --- |
| Tortoise SVN | 1.8.7 | For source code control | <http://tortoisesvn.net/downloads.html> |
| Astah Professional | 6.1 trial | For architecture design |  |
| Diagram Online Software | Free | For class design, Entity Relationship Diagram | http://creately.com/  http://websequencediagrams.com |
| MS Office | 2010,2013 | For Documentation |  |
| Snagit | 12 | For Document | http://www.techsmith.com/snagit.html |
| Sublime Text | 3 (Trial) | For development | <http://www.sublimetext.com/3.html> |
| MySQL | 5.6.15 | For development |  |
| XAMPP | 1.8.3 | For development |  |

## CI Baseline Procedure

#### For Document



#### For Source code:

No

No

Found

Found

Developers code in Local path then check in to SVN for review and integration

Code is reviewed and integrated in SVN then moved to **WIP/Source** with suffix **Untested** if passed review and Unit Test

Tester get source from **WIP/Source** to execute Integration Test and System Test

CIs are check-in to **WIP/Source** with suffix **Tested**

Developers get latest source code from SVN

Developers get last test source from SVN and work on change, if have

## Project Baseline schedule

| No. | Baseline Name | When Baseline | PIC |
| --- | --- | --- | --- |
| 1 | Startup | Within 7 days from approval. It is mandatory requirement that version of all CI at Startup baseline to be archived in separate folders in Archive area | Project team |
| 2 | Solution | When Architectural design v1.0 is released and baseline | Project team |
| 3 | Construction | Right the end of development phase | Project team |
| 4 | Wrap-up | After the final release. It is mandatory requirement that version of all CI at Wrap-up baseline to be archived in separate folders in Archive area | Project team |

## Directory structure & Access right

### Promotion Area

| Area | Purpose |
| --- | --- |
|
| Develop Area | Area for different users to store his/her owned items |
| Review Area | To store items that is ready for review.  Reviewer get to be-reviewed items from this area |
| Test Area | Just applicable for Source items.  To store items passed Unit Test and Code Review |
| Release Area | To store the items ready for release and all released versions of items  Users get the most recent items for their usage from this area |
| Archive Area | To archive all released versions of each CI  Archive area is a protected area for project baselines where all the CIs cannot be changed by any member |

### Directory structure

| Main Folder | Sub Folder | Purpose | Map to Area | Access right |
| --- | --- | --- | --- | --- |
| Project Directory : | | | | |
| WIP | Documents | Documents of Requirements, Design, Test, … | Release + Review | Full: PM, CC  Modify: PIC  Read: All |
| Deliverables | Document of Reports (1->6) to deliver | Release + Review | Full: PM, CC  Modify: PIC  Read: All |
| Meeting minutes | Store project meeting minutes, including meeting minutes with customer | Release + Review | Full: PM, CC  Modify: All |
| Plan | Store Proposal, Estimation, Project Plans, Project schedule, Task list | Review + Release | Full: PM, CC  Modify: PTL  Read: All |
| Record | Store project records, divided into  Review: include Review, Test and Inspection records  Change request  Acceptance  Mails  ... | NA | Full: PM, CC  Modify: All |
| Source | Store VSS file of Source code | Archive | Refer to VSS directory |
| User | User’s working area, store user’s owned items | Develop | Full: PM, CC  Modify: User  Read: All |
| Reference | Process | Store Documents and Other materials/data supplied by customer or those support software development and production operation in the project… | Release | Full: PM, CC  Modify: PIC  Read: All |
| Template | Store Guidelines/Standards/Forms/Templates/Checklist specified for the project usage | Release | Full: PM, CC  Modify: PIC  Read: All |
| Final |  | Final document | Release | Full right: Project QAs  Read right: All |

#### Access right control

Access right of non-project team members (ex: auditor, external reviewer, etc…) must be get permission of PM and granted in the pre-defined duration, then revoked at expiry date by CC. As soon as a member is out of the project, his or her access right is revoked also.

The access right is reviewed frequently and updated by CC at <baseline point and project closure time>.

After project asset is approved by QA at project closure time, PM informs to IT Department to revoke the access right of all project team members. If someone has a request for data reference, audit, etc…, he or she must get the approval of authorized person, normally Group Leader or Division Leader, and then send the request to IT Department. IT Department is responsible for implementing such kind of requests.

## Version numbering rule

#### For Documents:

The version level is maintained as numbered identifier with two components

1.1

Revision

Version

The original version will be numbered 0.1. Subsequent revisions will be numbered 0.2, 0.3. The release version will be 1.0.

**Version number**, which appears to the left of the decimal. It changes only when the core architecture of the item changes. For example: when an item is completely overhauled, with substantial internal changes, the version 1.0 would become version 2.0

**Revision number**, which appears to the right of the decimal. It changes when existing content is changed, but the overall structure and flow of the item remains the same. The normal sequence of revision is 1.1, 1.2 and so on.

#### For Software Source Files:

Software executable and support files are generally identified by name and version number, such as “Main DB v1.1.a”. The scratch edition will be 1.0. The version numbering scheme consists of three components:

1.1a

Revision

Version

Update

view

**Version number**, which appears to the left of the decimal. It changes only when the core architecture of the software item changes, as when moving from one area of the development tool to another, when an application is completely overhauled, or the user interface changes fundamentally. In this case, version 1.1a would become version 2.0.

**Revision number**, which appears to the right of the decimal. It changes when new features, functionality or other content are added or significantly changed. In normal case, the core architecture or user interface have been extended or limited in some manner. The most common reason for changing the revision number is when adding a new module or other functionality to the software. The normal sequence of revision is 1.0, 1.1, and 1.2 and so on.

**Update level** is appended or incremented when the only change to the software item is to correct one or more defects, without the addition of any new functionality. Version 1.1 would become v1.1a, v1.1b and so on. This updating is over ridden when a combination revision, involving bug fixes and new feature additions, is performed. In such a case, the software revision number is incremented and any update indicator is dropped, as in v1.1b to v1.2

## Other CM rules

**Rule for changing document**: When a member wants to develop or modify documents, he must copy the documents from Documents folder to User folder to work on the change (if any), then merge it to Document folder with another version. Members are absolutely not allowed to edit directly on the Project folder. Rule for development: Refer Coding guideline