Configuration Management

## Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Detail | Author |
| 07/08/2014 | 0.0.1 | Initial Draft | James Singh |
|  |  |  |  |

# Nomenclature

* Folders will be named with self-explanatory names, and will be represented as square bracket enclosed name, eg. [CM Plan]
* Files will be named with self-explanatory names prepended with project id, and appended with version number (versioning approach defined below).
* Names can be dash-separated but not spaces or underscores. Underscore will separate various parts of the filename, such as project code, file name and version.  
    
  Eg. PRJ-2014-101\_User-Manual\_0.0.1.doc  
  Here,  
  PRJ-2014-101 represents project code (In place of /, dash is used)  
  User-Manual represents the name of the file, and its purpose  
  0.0.1 represents file version

## Folder Structure

[Configuration Management Plan]

[Development and Testing Plan]

[Testing]

[Test Cases]

[Unit Test Cases]

[Integration Test Cases]

[Test Plans]

[ITP]

[UTP]

[Knowledge Management]

[Meeting Notes]

[Team Folder]

[Mojahidul]

[Weldu]

[Talakisew]

[Yogendra]

[Va]

[James]

[PM Plan]

[Project Charter]

[Budget]

[Issue Management]

[Post Project Closure]

[Communication Plan]

[Team Meetings]

[Project Change Management]

[Risk Management]

[Project Plans]

[Archive]

[Iteration Plan]

[Scope Detail]

[WBS]

[Resources]

[User Manual]

## Versioning

File versioning

<Major Release>.<Minor Release eg. milestone>.<Revision>

Eg.

Prior to any release: 0.0.1, 0.0.2, 0.0.3 …

First minestone: 0.1.0

After first milestone: 0.1.1, 0.1.2, 0.1.3 …

Second milestone: 0.2.0

After second milestone: 0.2.1, 0.2.2, 0.2.3 …

...

First project release: 1.0.0

Therefore, files will be named like:

**PRJ-2014-101**\_**Change-Request**\_**0.1.2**.doc

File version, and revisions will be maintained at the top of the file as below:

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Date | Version | Detail | Author |
| 07/08/2014 | 0.0.1 | Initial Draft | James Singh |
|  |  |  |  |

Code versioning

Semantic Versioning will be used. Versioning will be done as:

MAJOR.MINOR.PATCH

Eg.

Prior to any release: 0.0.1, 0.0.2, 0.0.3 …

First milestone: 0.1.0

After first milestone: 0.1.1, 0.1.2, 0.1.3 …

Second milestone: 0.2.0

After second milestone: 0.2.1, 0.2.2, 0.2.3 …

...

Upon project completion: 1.0.0

*Release candidate:*

Release candidates are the versions that are awaiting release, and will be represented as RC. ALPHA and BETA represent alpha and beta releases respectively. They will be versioned as:

1.0.0-alpha < 1.0.0-alpha.1 < 1.0.0-alpha.beta < 1.0.0-beta < 1.0.0-beta.2 < 1.0.0-beta.11 < 1.0.0-rc.1 < 1.0.0.

## Audits

There will be audits to ensure compliance and adherence to quality.

Security Audit (Information security, hardware security)

Risk mitigation plan audit

Configuration Plan Adherence audit