ANOTODE

The Web Annotator .

Software Configuration Management Revision 2.0

CS Group 1

Authors:
Saurabh N (201451017)

Reviewers:
Mukesh S (201451060)
Abilash G (201451031)

Revision Table

Revision	Author	Reviewer	Revision Date	Revision Tracking Notes
1	Saurabh	Abhilash	12-10-16	add SCM activities
2	Saurabh	Mukesh	02-11-16	add info about version control

Contents

1 Introduction **3**
- 1.1 Purpose . . . 3
- 1.2 Scope . . . 3
- 1.3 Reference . . . 3

2 Configuration Management **3**
- 2.1 Application Policies, Directives and Procedures . . . 4

3 Software Configuration Management Activities **4**
- 3.1 Identification . . . 4
  - 3.1.1 Description . . . 5
- 3.2 Version Control . . . 5
  - 3.2.1 Description . . . 5
  - 3.2.2 Increasing Version Number . . . 5
- 3.3 Configuration Control . . . 6

4 Resources **6**

5 Plan Maintenance **7**

1 Introduction

1.1 Purpose

The purpose of this document is to identify and describe the overall policies and methods for the Configuration Management activities to be used during the life-cycle of project Anatode. The Configuration Management Plan (CM) identifies the items to be managed, the methods by which they are managed and the control processes necessary for coordination and control. The primary intention of this CMP is to provide overall information on the Configuration Management policy and methods to be adopted and implemented for the project including: submitting, planning, approving, and implementing necessary changes.

1.2 Scope

It is a firm objective that this Configuration Management Plan (CMP) shall apply through all the system life cycle phases of the project which began with the requirements phase through the production and maintenance phases and to the systems retirement phase.

Appropriate baselines are established at the start of the development phase and are applied to each Configuration Item.

1.3 Reference

1. IEEE Std 828-2005 (Revision of IEEE Std 828-1998) IEEE Standard for Software Configuration Management Plans.

2 Configuration Management

Configuration Management identifies what is required, designed, and produced. It also provides for the evaluation of changes including effects on technical and operational performance. This leads to making the configuration visible and understood by all the parties involved with the project.

2.1 Application Policies, Directives and Procedures

1. Team will make constructive criticisms with suggestions for improvement
2. Team members shall accept responsibility and accountability along with the authority given.
3. Code will be open source and no copying without mentioning credit.
4. Author of any document shall take the help of documentation team and draft the document in right format by himself and handover to reviewer.
5. In any conflicts and arguments between teams/group members will be entertained by group leader, and PM if not solvable by group leader.
6. Documents related to software development shall be kept stored on google drive and related code to be shared on github.
7. Teams shall use slack channels for communicating and coordinating with each other.

3 Software Configuration Management Activities

The main purpose of SCM is to make, report and track any changes made to the original software development plan. It is applied throughout the software development process and will help us to keep track of changes and also help us go through and make changes. SCM procedures will give us a good map out of the software so that if we are need to make more changes it will be relatively easy to do so. SCM will maximize productivity by minimizing mistakes. For SCM to be successful, all the members of software production team will have to take time to report the changes that they think are necessary and/or to notify others of changes that they may have made. This is sort of boring and time-consuming work, but it is very important.

SCM activities are developed to

1. Identify change
2. Control change
3. Ensure that change is being properly implemented.
4. Also have a way to document the change.

3.1 Identification

In this section we will describe the way software configuration items will be identified for the software configuration management plan.

3.1.1 Description

1. **Identify change**
   If during the software development phase a team member suggests a change in the software then we need to have the team work on the suggestion and to figure out if the change is necessary and is justified.

2. **Approve change**
   We want to be able to have control over any change within the software. We can not afford to have one member of software development team think of a change or and implement it without telling any other member of the team. This can create huge technical problems for the software. We want to develop rules so that no member of the team will think of and implement change without permission of other members. We will be using the change request report form to suggest changes in the software.

3. **Ensure that change is being properly implemented**
   We want to have team members looking over the change. Since all the teammates will be working separately, it is possible to have made mistake in implementing the change. To make sure this doesnt happen, we want to set up times when team members will look over the change that other members have implemented and to finalize the change.

3.2 Version Control

3.2.1 Description

As a result of changes, the version number of various modules will be increased accordingly. We will be using a universal version number system for all modules. We will also have a final version of the entire product. Major documentation will also have version numbers, such as User Manual or Design Specification.

3.2.2 Increasing Version Number

When a change request is filed, a change report will be created. After the change is finalized, it will be documented in the library. We will be using a decimal point version number system:

major update . minor update bug fix

Bug Fix
If enough bug fixes have been done on the product/module, the bug fix portion of the version number will be increased. The number of user visible bug fixes will also affect when the bug fix number is increased. The more visible bug fixes have been made, the closer

the bug fix number will need to be increased.

Minor Update
If functionality is added to the product/module that will increase the user-friendliness / performance but does not change the way a function/interface work, the minor update number may be increased. Several of these changes will warrant a version number change.

Major Update
We do not foresee any change in major version number. The product will be labeled as version 2.

3.3 Configuration Control

Changes will be controlled by using human procedures and automated tools. Here are the steps, which will be taken in order to control change.

- Request the change
- Software developer will evaluate the change request.
- The result of the evaluation will be presented as change report.
- Final decision on change will be made.
- If change is approved
    1. Define constraint
    2. Check out items for changes
    3. Make necessary change
    4. Apply testing activities
    5. Rebuilt the software

4 Resources

- Git is a free open source, distributed version control system designed to handle everything from small to very large projects with speed and efficiency.
- Google Drive is an online file storage and synchronization service created by Google. All the documents are stored here for the easy access by the team members.

5 Plan Maintenance

The Plan should be reviewed at the start of each project software phase, changed accordingly and approved and distributed to the project team. Changes made to Software Configuration Management Plan are evaluated and approved by the review team. When the review team reviews and modifies the Software Configuration Management Plan, the changes made are communicated to all the team members and the new Software Configuration Management Plan is shared with all the team members.