CAMPUSTALK

Configuration Management Plan

**Team-5**

**IT-632 Software Engineering**

**Instructor- Prof. Asim Banerjee**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Author(s) | Date | Description |
| 1.0 | Sunny Lalwani |  | Created |
| 1.0 |  |  | Reviewed |

**Contents**

**1. Introduction**

1.1 Purpose of document

1.2 Scope

**2. Configuration Management**

2.1 Approach

2.2 Organization

2.3 Tools used

**3. Configuration Management Activities**

3.1 Configuration identification

3.2 Configuration control

3.3 Configuration status accounting

3.4 Configuration audits

**1. Introduction**

The Configuration Management exercise is one of the most important techniques under Software Engineering, as experience has shown that creating extensive software as a team involves self-discipline. Individual initiatives, when requested and co-ordinate, give successful results. The configuration of software gives the state program programs are in. All the factors like work products, source requirements, test cases etc express the position of program. All the records created during the life-time of program need to be put under tight guidelines of access, adjustment and adjustment to ensure that peace of mind dominates and there is no disorder. With this view in mind this papers provides our configuration management Plan.

* 1. **Purpose of the document**

The purpose of this plan is to describe the criteria and direction necessary to perform the activities of software configuration management. It defines the steps and activities that describe how Configuration and Change Control Management is performed in the development of a software product.

The basic reason of software configuration management is to reduce the rework required for the changes and make the software consistent in order to meet the cost, quality and schedule objectives.

**1.2 Scope**

This document provides the direction and guidance for the Configuration Management during the software development life cycle. It pertains to the product expected by users and ensures the quality of the products developed at the end of project. It also establishes the configuration management methods used during the development and maintenance of our Project.

**2. Configuration Management**

**2.1 Approach**



Change methodology steps followed by the software:

* ***Initiate change:*** Anyone who is involved with the project can initiate a change through the submission of a *change request* (CR) ensuring that change requests are created with *consistent quality* and completeness and discards irrelevant requests.
* ***Analyze impact:*** After the arrival of change request, the review from developers and the team leader will be taken to determine the *impact on existing documentation* and configuration items e.g. requirements, architecture documents and/or software configuration items.
* ***Authorization of change request****:* We have established a formal *process for authorizing* change in which the change management team review the change request and *vote on the changes* according to predefined voting logic (i.e. two- third majority). If the change request does not get the two-third majority the change request is discarded.
* ***Develop change:*** After Authorization of change request the developers make sure that the change meets *acceptance criteria* before it is passed to the release management process.
* ***Release the change:*** *Plan production releases* resulting from approved change requests. The coders will take care of minimum disruption to the production environment. After releasing the new version, which includes the change request we will further prepare for review to ensure maximum successful deployments.
* ***Close the CR* and inform the initiator.**

This process will continue until the final document satisfies all the members.

**2.2 Organization**

Roles and responsibilities:

|  |  |
| --- | --- |
| Team Members | Roles |
| Anyone from the team | Initiator |
|  | |  | | --- | | Configuration manager- work product | |  | |
|  | |  | | --- | | Configuration Manager- Source Code | |
| Aresa Vora | Team Leader- Monitor the Team |

**2.3 Tools Used**

**Version Control of Documentation**

Since documentation of our project is being collaborated by multiple members of the group, we’re using multiple applications to jointly create documents, that include Google Docs (which provides real-time multi-user editing) along with Dropbox for syncing and versioning file changes as and when they happen. Also, each time major changes occur to the document, a new numbered copy of document is created so that it becomes easier to track changes.

**Source Control Management**

We’re using Git Version Control system for all the code that we have in our project. It is a distributed version control system and is best suited for projects which are developed in teams where multiple contributors work on the same component. It allows us to create branches based on feature set or user and there’s a **master branch** that which contains main source code and all the branches are created from and merged to the master branch. All the changes that occur in the code are **committed** and hence commits are the actual mile-stones of everything that has changed since creation of the source code and thus it is possible at any point in time to revert back to a state of code based on commit. In The IDE we’re using for development, Eclipse, has tight integration with Git and makes it easier to manage branching and commits using GUI.

**3. Configuration Management activity**

The CampusTalk Configuration Management process further defines the need to trace changes, and the ability to verify that the final delivered software has all of the planned enhancements that are supposed to be included in the release. The process used to manage the CampusTalk’s configuration items fall in four main categories:

1. Configuration identification

2. Configuration control

3. Configuration status accounting

4. Configuration audits

**3.1 Configuration Identification**

Configuration identification involves identifying which items are to be placed under configuration management. All the development artifacts like source code, software tools, documentation are identified as configuration items. The software product and corresponding software elements should have a unique identifier such as a name and version number. Configuration identification is a pre-requisite for the other configuration management activities, which all use the products of configuration identification.

Identified configuration items can be classified in four classes:

**1. Evolving items:**

Project plan, SRS document, design document, test plans, test report, coding conventions, user manual, Quality Assurance Plan, Configuration Management document.

**2. Source items:**

Source code

**3. Support items:**

Applications, Development tools, Apache server, MY SQL

**4. Archive items:**

Time sheets

**3.2 Configuration control**

Following are the rules and methods to be followed:

1. Every product before extracting to be worked upon by a developer should be baseline.
2. Each developer must save an original copy of the artifact and make his/her respective changes in a separate folder on his/her Dropbox.
3. Each developer can own a copy and make changes as per he wants, check the results at his local host, but should never overwrite the previous version. He/she must upload the modified document/code by the name documentName\_version

**3.3 Configuration Status Accounting**

The purpose of Configuration Status Accounting (CSA) is the recording and reporting of information needed to manage configuration items effectively, including:

1. A record of the approved configuration documentation and identification numbers.
2. The status of proposed changes, deviations, and waivers to the configuration.
3. The implementation status of approved changes.
4. The configuration of all units of the configuration item in the operational inventory.
5. Discrepancies from Functional and Physical configuration audits.
6. The record of configuration data and to present a coherent picture of the product at determined moments in the product’s life-cycle

**3.4 Configuration audits**

Configuration auditing is the process of verifying the configurations of assets to ensure they match with stated security and compliance policies.

Configuration auditing provides three key benefits:

1. **Lowers the cost of compliance** by reducing the time and cost to prepare for audits with increased confidence that assets are properly configured.
2. **Increases security with actionable insights** with the ability to quickly and effectively establish a proper and secure baseline for the configuration of all assets and monitor incremental changes.
3. **Reduces burden and increases efficiency** of operations with continued visibility and insight into the environment, including changes and new introductions (planned and unexpected), with meaningful and actionable reports.

The point is to select a solution that provides insights that reduce the cost of audit preparation while also improving security and assuring the configurations of assets on an ongoing basis.