MIDDLE EAST

TECHNICAL UNIVERSITY

COMPUTER ENGINEERING

DEPARTMENT

Configuration Management Plan

- Myrzabek MURATALIEV – *1408665*
- Serdar DALGIÇ – *1448570*
- Haşim TİMURTAŞ – *1449149*
- Barış YÜKSEL – *1449826*

Table Of Contents

1. INTRODUCTION

1.1. Purpose of CMP

"Nothing endures but change" says Heraclitus, the famous philosopher of Ephesus (c.535 BC - 475 BC). In most software projects, the success is dependant on the awareness of how important covering the changes is. With the awareness of the fact, we prepared this document.

It's an inevitable feature of a software development project that the project contains changes, updates, version bumps, releases and etc. All of these aspects should come out with a proper way and methodology. Another thing is, software projects are developed by a number of people. That's why a tracking plan is what prevents the project to turn into a huge mess. Any change or update which is done by a member must be documented and all of the team members must be informed of these changes to prevent any possible inconsistency. Therefore, planning is necessary to handle such changes and updates.

Our purpose of preparing a Configuration Management Plan for our Project is managing the configuration of the “Wekarel” Project throughout its lifecycle, which is conducted by four of us. Identification, management, control and auditing of the changes and updates will be decided during the preparation of this document.

1.2. Scope of Document

This document is prepared to supply the relevant policies and procedures for the development of Wekarel. It describes the organization and responsibilities for configuration management of Wekarel and the Configuration Management(CM) process which includes the identification and management of Configuration Items (CIs), version controlling and audit-changing. Moreover, we will declare the milestones and schedule(current status of our Living Schedule), resources we are going to use and the plan optimization strategies. In other words, we describe all

Software Configuration Management(SCM) activities and terminologies in this document.

1.3. Definitions, Acronyms and Abbreviations

CM *Configuration Management*

CMP *Configuration Management Plan*

CI *Configuration Item*

CCB *Configuration Control Board*

TT *Testing Team*

DT *Development Team*

SCR *System Change Request*

SVN *Subversion*

WEKA *Waikato Environment for Knowledge Analysis*

1.4. Document References

- IEEE St. (IEEE Std 828-1998) for Software Configuration Management Plan

  http://standards.ieee.org/reading/ieee/std_public/description/se/1042-1987_desc.html

- SciComp Official HomePage

  http://senior.ceng.metu.edu.tr/2009/scicomp/

- Software Configuration Management – METU Computer Engineering CENG 492

  http://ceng.metu.edu.tr/courses/ceng490/

Also this document is in convenience with the previous documents of the Wekarel project such as Analysis Report and Detailed Design Report.

1.5. Document Overview

This configuration management report is composed of 6 parts that are determined with respect to IEEE standards and “Software Configuration Management” presentation prepared in METU Computer Engineering Department for the course CENG492.

In the *Introduction* part, the purposes and the need to write a CM Plan is explained and which scope does this CM Plan exists. Definitions and abbreviations for the concepts commonly used throughout the CM Plan are also mentioned in this part with the related Document References.

In the *CM Framework Organization* part, the organization among the group members and their responsibilities are explained. In addition to this, tools going to be used are stated with a few sentences.

The CM Process part briefly explains the identification, management, and the auditing of the project.

The important dates, milestones and the deadlines are explained with their effects are mentioned in *Project Schedule – CM Milestones* part.

Project Resources is the next topic and explains furthermore resources that are going to be used for CM activities.

And the last; *Plan Optimization* tells us the methods which can be used to optimize the CMP.

2. CM FRAMEWORK ORGANIZATION

2.1. Organization

SciComp is a small company with four members and if we visualize the company’s general functioning, it is seen that there is not any activities with responsibilities which demand existence of corresponding organizational units as a

difference from big companies. Therefore, our company has small number of units with relevant responsibilities which come together to be adequate to monitor the Software Configuration Management activities. Additionally, every person of this project will be member of all units. This decision aims to force the members to be updated and share the responsibilities. Assignation of various roles in development are planned to be held in company's weekly meetings according to the state of the project. So, our first and important unit is Configuration Control Board(CCM). Second and third units will be Developing Team(DT) and Testing Team(TT), respectively.

2.2. Responsibilities

As every person is a member of all units, responsibilities of all units will propagate to all members. Apart from that each individual will have routine responsibilities to avoid possible hindrances in proper functioning of company.

Individual responsibilities of each member:

- Informing the members about SCR by sending e-mail to the groups
- Writing proper comments about what they are changing when committing to SVN
- Conforming to the CM schedule

Responsibilities of CCB:

- Make decisions on System Change Requests(SCR)
- Coordinating roles among team members
- Documenting, controlling software baselines
- Updating CM schedule
- Auditing

Responsibilities of DT:

- Implementation of SCR’s
- Creating baselines and releases
- Documenting the releases

Responsibilities of TT:

- Testing and debugging after SCR implementations
- Documenting user guides

2.3. Tools And Infrastructure

2.3.1. SVN

As a version control system, Wekarel will use SVN. Features of SVN:

- In commit operations, atomicity is preserved . Whenever a commit operation is interrupted, there will not be an inconsistency or corruption in the repository.
- All files committed in the repository exist in the revision history even if they are modified, removed, etc.
- Directories, renames, and file metadata are versioned.

2.3.2. SFTP

In order to reach the source code under the repository, we need to use secure file transfer protocol (SFTP). Also, we use it to transfer the source code with PBS script from the local directory to compile and execute on NAR system.

2.3.3. SSH

In addition to sftp, to establish a secure connection with NAR, ssh is our key tool to achieve this goal.

2.3.4. Trac

Trac is an enhanced wiki and issue tracking system for software development projects. It provides an interface to Subversion (or other version control systems), an integrated Wiki and convenient reporting facilities. These feature make *Trac* our most helpful guide in tracking the development of our project with SVN.

2.3.5. Eclipse

As an integrated development environment, we will use Eclipse. WEKA developers suggest IDEs like Netbeans and Eclipse while developing extra features for it. Paying attention to that fact, Eclipse is decided to be our development platform for writing code snippets in the project. Moreover, it makes our work easier that Eclipse support for both Windows and Linux systems are available.

3. CONFIGURATION MANAGEMENT PROCESS

3.1. Identification

3.1.1. WEKA source code written in Java

Our basic source code will be WEKA's source code. As stated in our design reports, we are going to write the parallelized versions of previously selected code parts and make changes due to our schedule. Modularity of the code pave our pathway and we are fine about the modularity of the original WEKA code. We will be obedient to original code's structure and the parallelized versions of the original classes are going to preserve their positions in the same directory with the originals.

3.1.2. Documentation

Although being crucial, documentation is ,most of the time, skipped over to pay more time to development process. But, as a futuristic point of view, documentation is the rope that ties the future with the yesterday of the project. That's why we pay more attention to documentation for our project to be succesfully developed in the upcoming days after we graduate. Some of the documents that we have already published are:

- Project Proposal
- Requirements Analysis Report
- Initial Design Report
- Final Design Report
- Living Schedule
- Configuration Management Plan

In addition to these, in the latter stages of the project, we plan to create and publish these documents too:

- Test cases of *Wekarel* and their documentation
- User documentation added to *Wekarel*
- Developer documentation for writing parallelized versions of the algorithms

3.2. Management and Control

Configuration control activities of CA request, evaluate, approve or disapprove, and implement changes to baselines CIs in this part of the CMP.

3.2.1. Change Request

During the implementation of the project, changes and modifications of the modules can be requested by the members, instructors and assistants. For these

requests, we use our wiki and/or mailing group. As we are a group of five, team members can communicate through phones or google group mailing list for such a SCR; but for larger projects and larger teams, systematic is as follows:

- ID of the SCR
- Date of the SCR
- Severity of SCR
- The component or the module of the SCR
- Deadline of the SCR
- Version to apply the SCR
- Description of the SCR
- Assignment of the SCR (default=”SciComp”)

3.2.2 Evaluating the Changes

When a change request comes, every member takes a look at the properties of the SCR, if there exist a misinformation or wrong information. After passing this part, the team members decide about the severity about the SCR. Strong communication between team members is essential in this part.

3.2.3 ACK/NACK (Acknowledging or Not-Acknowledging) the Changes

Acknowledging a SCR is approved by CCB, that includes of all the four team members. All of the members should acknowledge the SCR in order to be accepted from CCB.

3.2.4 Implementing the Changes

After SCR is accepted, at least one of the team members are assigned to this issue, and investigate the possibilities of the implementation of the SCR. The best solution is decided by CCB, and takes place in the updates living schedule plans.

3.3. Configuration Status Accounting

CSA includes activities for recording and reporting the CIs of the project. All the group members, instructors and assistants are informed about the status of the project by CSA. The related changes are mentioned every week in meeting reports and weekly meetings. In addition to reports and meetings, we will use SVN revision control system that gives chance to see the latest changes. Moreover, our website manages the informing business. When a source code is changed and committed into SVN, it must be commented by the related user. In fact, google group's mailing list and wiki help us to inform these changes. In the description part of the change, the reasons of the change and how the problem is solved should be stated. We have to observe if there is any inconsistency or not. If something goes wrong, we should be able to see which member is responsible for this situation. Finally, the changes must be numbered as versions and dates. SVN automatically does this after each commit.

3.4. Auditing

Auditing is vital for the success of a project. People tend to do unconcious auditing almost everyday. Coffee machines are a good place for such little auditings. We believe in the power of such little auditings. It's important to take these auditings physically, rather than online. Moreover, every week we make a big audit on Thursday where we discuss the project in general. The first thing that we consider in main audits is to achieve the design goals for the release. All the project members will participate for audits and reviews. The results of these audits will be stored in our SVN repository or our Wiki.

4. PROJECT SCHEDULES – CM MILESTONES

	Date	*Milestone*	*Additional Information*
1)	09.03.2009	CM Delivery	Our initial CMP
2)	20.03.2009	First Dev. Snapshot Demo	-
3)	26.04.2009	Project Test Specifications	-
4)	10.05.2009	First release, Demo	Modificaton of CMP according to the demo release
5)	19.06.2009	Final demo	Our final CMP according to the final release

5. PROJECT RESOURCES

At the first step, SciComp Members are the main CM resource of the *Wekarel* Project.

Apart from the tools we have explained in the tools & infrastructures (SVN, SSH, SFTP, Trac, Eclipse) we will use SciComp Google group for intercommunication between the group members, and the Weka mailing list for further possible updates, bugfixes, questions and help concerning the Weka tool.

Another resource will be *Wekarel* Web Page, where our living schedule, screenshots, benchmarks and project documentations can be visited.

6. PLAN OPTIMIZATION

Plan Optimization will be carried out with respect to the updates in CM plan, but it is not in a high priority position for our team as weekly meetings are carried out at least once a week. Communication between team members are achieved through e-mails, mobile communication and instant messengers; If any of the members thinks an optimization in CM plan is needed, extra meetings are to be arranged.

The CMP will be updated during the software lifecycle whenever required. The CCB will control if an update in the plan or the CM schedule is required or not.