CENG492 Configuration Management Report

Buğra Şirin
Deniz Tav
Engin Fırat
Mert Özer

BiBER

March 20, 2011

Table of Contents

1. Introduction

This document is the configuration management plan of Mindolog project, performed by the group Biber. Mindolog is a project which aims to provide an environment for psychologists and patients to fix attention deficiency and performance disorders by using neurofeedbacks provided from Brain Computer Interface Devices (BCID). It is a project which has a game playing part, a database and functionalities based on the saved and received data.

1.1. Purpose of Configuration Management Plan

Mindolog is a project in which many different modules work together and communicate with each other as explained in the Detailed Design Report. In other words, it is a system in which a number of components are integrated to perform the intended actions. Because of this interaction, all other developers should be notified when any modification made by a developer on a single entity. If not, the entities from separately developed modules would not meet the requirements needed for another module's functionality, which must be avoided.

During the development of software like that, updates and changes may occur at any time or any type of project material: source code, development plans, release versions, etc. This situation is inevitable, expected and normal, clearly; however, it is important to emphasize that those updates and changes are very critical for the project. In order to provide strong communication among developers and to improve quality of the product, configuration management is absolutely essential for such a stable development process, considering the fact that our modules, classes and implementation can always change in time.

1.2. Scope of the Document

The scope of this document is the identification of configuration management plan for the project. The plan presents configuration management activities that will be applied throughout the development process of our project and the activities explained in this document are applicable during the development and maintenance of the project.

The document contains methodologies and activities designed to manage changes and relate them. Also versioning, differencing work product and reporting on all are included in this document. The related audience for this document is Biber software team, our assistant and instructors.

1.3. Definitions, Acronyms and Abbreviations

- **BCID:** Brain Computer Interface Device
- **CM:** Configuration Management
- **CMP:** Configuration Management Plan
- **CMR:** Configuration Management Report
- **CI:** Configuration Item
- **SDT:** Software Development Team
- **TT:** Testing Team
- **RCT:** Release Control Team
- **CMT:** Configuration Management Team
- **CCT:** Configuration Control Team
- **SCR:** Software Change Request
- **CSA:** Configuration Status Accounting

1.4. Document References

- IEEE Standard for Software Configuration Management Plans (IEEE Std 828-1998).
- Pressman, Roger S. Software Engineering: A Practitioner's Approach, Sixth edition. New York, NY: McGraw-Hill.
- “Software Configuration Management”, the presentation prepared in METU Computer Engineering Department for the course CENG492.

1.5. Document Overview

In the introduction part we explained what is the meaning and purpose of a CMR and gave the abbreviations and the references used. The organization of the team, the responsibilities of team members and tools used during the development can be found in the 2nd section. The Configuration Management Process section, which is the 3rd section, gives information about the identification process, tools and practices for management and control, configuration status accounting and auditing. Project schedule and milestones are given in the 4th section. The resources needed are explained in 5th section. Finally, we conclude with the methods to be followed for optimizing Configuration Management Plan.

2. The Organizations CM Framework

2.1. Organization

All the members of Biber contribute to CM to develop a successful project and have equal rights during the project management process. Each team member is responsible for criticizing, giving ideas, being a part of brain-storming events. However, since there should be some management constraints about the process steps and configuration management, some sub teams are arranged.

- Software Development Team (SDT): All group members.
- Testing Team (TT): Buğra Şirin, Engin Fırat
- Release Control Team (RCT): Deniz Tav, Mert Özer
- Configuration Management Team (CMT): Buğra Şirin, Deniz Tav
- Configuration Control Team (CCT): Mert Özer, Engin Fırat

Responsibilities of the whole group and of each team are explained in the next section as well as the communication and links between the sub teams.

2.2. Responsibilities

Since each member of Biber is also a member of SDT, some responsibilities are taken by all the members;

- Conforming CM schedule
- Commenting about possible changes before committing resources through SVN
- Giving feedback about requested changes
- E-mailing group members about SCR
- Applying the possible changes before a given deadline

Besides, there are main responsibilities of each sub team:

- SDT is mainly responsible for implementing the modules of Mindolog project and making the changes that are requested by TT. This team also integrates the implementations with the rest. Lastly, releasing a version is also a duty of this team.
- RCT controls the current and next version, also gives feedback to SDT about the current release.
- TT will be responsible for two main duties. First duty is the testing and debugging of the implementations produced by SDT and to check whether the modules meet the requirements. Second one is checking if the requirements correspond to the implementations, stating the problems and giving feedback to SDT.
- CMT is responsible for maintenance of the CM organization. Hence, this group will keep the CMP up-to-date.
- CCT will supervise all the activities of other groups. However, the main responsibilities of this group are to review SCR's, accept or reject SCR's and monitor them.

2.3. Tools and Infrastructure

- **Java:** Because of the libraries provided of use, we are going to use Java programming language.
- **Eclipse IDE:** It is compatible with Java programming language, and with the library provided by our sponsor company Minder to use BCID. Therefore we have chosen this IDE to the design the system.
- **Emotiv BCID:** It is a brain computer interface device which gets the alphaand beta values with 50ms period (as default). Our sponsor company provides this device.
- **Emotiv Test Bench:** It is a program to see data coming from the BCID. The program controls whether the points that should give alpha/beta values is correctly placed to users' head. Moreover, it concurrently shows the incoming data.
- **jFreeChart:** It is a library to draw various types of graphs (line, pie chart etc.). It is available for Java programming language and distributed with GPL.

- **jMonkey:** It is s a game engine made especially for modern 3D development. The entire project is based on Java for high accessibility. Advanced graphics are supported through OpenGL 2 via LWJGL.
- **jopenGL:** It is similar to OpenGL in C/C++. It is extended from OpenGL for Java language. As mostly known, jOpenGL gives us a chance to design game environment.
- **PostgreSQL:** We will use PostgreSQL database management system. It is a powerful, open source object-relational database system.
- **swing:** It is a library to design frames, windows in given environment. In other words, it is a library to design GUI.
- **SVN:** This version-control system helps us developers to backtrack, maintain current and old versions of implementations. Plus, since Biber consists of four members who work on the project at different times and places most of time, SVN will satisfy the mobility of the project on different operating systems and hardware systems. Our department provides this tool and enough space for version controlling.
- **Trac:** Trac is an open source, web-based project management and bug-tracking tool.
- **Team Web Page:** All documents and project progress can be seen via web page.

3. Configuration Management Process

3.1. Identification

The configuration items are important to identify our project Mindolog and these items are divided into 3 categories: Source Code, Data, and Documents.

3.1.1. Source Code

The source code files are kept in the repository of the SVN hosted by Department of Computer Engineering, METU resulting in access to all members of the team adding, updating source files and committing afterwards. All of the implementation will be done in Java programming language. Coding phase is the most important part of the project and maintained by all the members of Biber. CI's of the source code are named according to its functionalities. Each module in Mindolog is a CI and these are explained below:

- Core Module: This module manages the system. Moreover, all the data flows are accomplished in this module. In this manner, other modules can communicate with each other.
- BCID Module: This module manages the data supply from the BCID.
- Game Module: Game uses the incoming data coming from the data flow managed by the core module. Game proceeds by these data and keyboard input.
- Graphical Interpreter Module: This module uses the incoming data coming from the data flow managed by the core module and uses these data to produce output graph which shows the process of the patient.

3.1.2. Data

Data consists of the main database and the log modules. The database is a CI as a whole and needed for storing information related to units used in Mindolog. Besides, according to incoming data from Core module, logger saves the error message to a log file. The main purpose of this module is to provide technical data when an unexpected situation occurs and this is also a CI.

3.1.3. Documentation

In our project, documentation is a mandatory CI. The documents are;

- Project Proposal Report
- Requirements Analysis Report
- Initial Design Report
- Detailed Design Report
- Revised Design Report
- Configuration Management Report
- Test Specifications Report
- User Manual
- Web page sources
- Comments on Source Codes
- Weekly Reports

3.2. Configuration Management and Control

If any member of our group believes that some change is needed at some point of the system, these points must be clarified;

- The reason of the possible change
- The parts of the system that will be affected from this change
- Which parts of the codes must be changed

If CCT approves it, requested change should be handled by corresponding team members according to affected modules. Then, there will be tests to make sure that whether the changes are applied or not. If the new design passes the test, changes will be reflected; however, if anything unwelcomed occurs, the old version would still be online at SVN.

3.2.1. Software Change Request

Minor SCRs will be discussed between team members in person and directly added to the system. These changes can be handled by SVN and requires no extra information.

Major SCRs are controlled by the Trac system. In this system a SCR consists of:

- Team member name
- Description
- Date
- Deadline
- Related Module(s)
- Priority
- Version

When a SCR reported, a ticked is opened by Trac and it can be seen by all the team members. Next, evaluation and possible implementation is discussed.

3.2.2. Software Change Evaluation

The discussions about evaluation of SCR are maintained on tickets in the Trac system. Moreover, all SCR's will be discussed face to face in team meetings and during the daily communications of the team members. In each evaluation or comment, all members can emphasize their opinions freely and evaluation will be based on these opinions.

3.2.3. Software Change Implementation

If SCR is approved after an evaluation, all the possibly affected CI's will be detected and the implementation part of that request is assigned to the group member(s) that has the responsibility of the corresponding module(s). After the implementation is finished, the member uploads the code to the SVN. Next, TT downloads the updated/added module and deploys it in their system to check if the change leads to any problem such as side-effects etc. Test cases will be tried to cover all the aspects of the implementation and if it passes the test, the change will be reflected. Otherwise, the members responsible for the implementation of the related change are informed and either they review their implementation or another member takes over the code segments to debug the implementation. During each step, every group member will be informed about the change.

3.3. Configuration Status Accounting

Accounting the status of the development at every little change is crucial in CMP. When a developer updates a source file in SVN, he informs all the other members and the TA as well about what he added or updated. Purpose and the description of the change are going to be explained clearly in the comments. All the members are responsible for taking care for a possible corruption or loss of data on the project. Also the reason of the change will be kept in a log clearly. This provides the development to be followed easily. Living schedule is updated as well in regards of the accounting mechanism.

3.4. Auditing

When a member completes a task and is about to upload his source file to the SVN repository, it is obvious that he first needs to make sure that the code he wrote is working and also he should have tested it to check if it works correct on trivial cases, after that TT takes over testing part.

The members of Biber periodically check each other's work via the Trac system and give feedback to each other personally when necessary. Project timeline determined is intended not to be changed.

The most important auditing factor is the exhaustive testing of system before the final release to be conducted by the developers.

4. Project Schedules and CM Milestones

The living schedule with all tasks and milestones to be completed are in the website of Biber. Good separation of components result in better functionalities, and also implementation of only that particular component makes it easier to update and change the system's functionalities. Having implemented the prototype at the end of the first term, implementation of better modules, new visuals, missing game functionalities, database and logger module are the major items in the list for the upcoming release.

The milestones of the projects are as following:

- **First Development Snapshot, Demo:** Modular version of the first semester's prototype is going to be exhibited. We are going to identify and add new modules upon the first semester's prototype. This demo will be exhibited in the week starting at March 28.

- **First Release:** It is an official milestone in order to finish the project. We are not going to make any architectural change in the project after the first release. This demo will be exhibited in the week starting at May 9.

- **Final Package:** It is the end of the project. We are going to finish all modules until then, and we are going to complete integration, testing and documentation. Project will be delivered in the finals week, end of May.

Apart from these major milestones, Test Specification Report will be prepared and delivered in the week starting at April 25.

5. Project Resources

The following tools are used for CM activities by our group:

- **SVN:** Revision Control System
- **TRAC:** Issue Tracking System
- **Eclipse:** Development Environment
- **Web Site:** The website of Biber is where the user resources are stored. All of the documents produced during the development of the project are stored in here
- Project Mail Group

6. Plan Optimization

CMP will be a guide for coordination and progress of Mindolog and any update or change in the CM schedule will be controlled by CCT. When an update or change is determined to be done, all group members will be notified via Trac.

There will be meetings regularly to keep track of progress of configuration items. According to these meetings, the living schedule will be updated regularly. These meetings will help us to continue development according to the predefined schedule; however, as in all software development processes, there can be some unpredicted problems. Obeying the schedule is a must for the optimization issues on CMP. Obviously, delaying a milestone date might occur naturally once in a time but the intention is to keep this at minimum. During the project, if there will be a need to change in CM plan, Biber members will meet and update the schedule accordingly.

Besides, since every milestone is supposed to be accomplished with the efforts of every single member, each member is responsible from himself and has to report to the rest of the team.