Configuration Management Plan

For

Online National Election Voting

Group: iTeam4

- Emilbek Joldoshev 1592476
- Hassan Salahe Matar 1591114
- Mehmet Barış Özkan 1560747
- Hüseyin Lutin 1560408

20/03/2011

1. INTRODUCTION 3
1.1. Purpose of Configuration Management Plan 3
1.2. Scope Of The Document 3
1.3. Definitions, Acronyms and Abbreviations 3
1.4. Document References 4
1.5. Document Overview 4
2. Management 4
2.1. Organization 4
2.1.1. Software Development Team 4
2.1.2. Testing Team 5
2.1.3. Configuration Management Team 5
2.1.4. Configuration Control Team 5
2.1.5. Release Control Team 5
2.2. Responsibilities 5
2.3. Tools and Infrastructure 6
3. CONFIGURATION MANAGEMENT PROCESS 7
3.1. Identification 7
3.1.1. Source Code 7
3.1.1.1. Election/Normal Mode Graphical User Interfaces 7
3.1.1.2. Model Logic Classes 7
3.1.1.3. Control Logic Classes 7
3.1.2. Database 8
3.1.3. Documentation 8
3.2. Configuration Management and Control 8
3.2.1. System Change Requests 8
3.2.2. System Change Evaluation 9
3.2.3. System Change Implementation 9
3.3. Configuration Status Accounting 9
3.4. Auditing 10
4. PROJECT MODULE AND CM MILESTONES 10
5. PROJECT RESOURCES 10
6. PLAN OPTIMIZATION 11

1. Introduction

1.1. Purpose of Configuration Management Plan

The purpose of this document is to define and explain the Configuration Management Plan for Online National Election Voting (ONEV) system. In this project several system components such as user interfaces, database and application server are integrated to each other and working of these components depend on each other. Because of that fact any modification done by a developer in one component should be notified by all other group members in order to maintain the development of the project. The Configuration Management Plan is prepared for solving this problem and sustaining the improvement of the ONEV.

1.2. Scope Of The Document

This document is prepared by the developers of the ONEV system. The scope of this document is about identification of configuration management plan for software developers of the ONEV project. Throughout the document the development process and activities that will be conducted according to CMP are described. Coping with versions mechanism, standardization, milestones, and resources are also in the scope of this document.

1.3. Definitions, Acronyms and Abbreviations

ABREVETIONS	DEFINITIONS
ONEV	Online National Election Voting
CM	Configuration Management
SDT	Software Development Team
TT	Testing Team
CMT	Configuration Management Team
CCT	Configuration Control Team
RCT	Release Control Team
SVN	Subversion
SCR	Specification Change Request
CSA	Configuration Status Account

Table1. Abbreviations

1.4. Document References

- Software Requirements Specification, by iTeam4, Fall 2010
- Detailed Design Report, by iTeam4, Fall 2010
- IEEE Standard for Software Configuration Management Plans (IEEE Std 828-2005)
- Software Configuration Management, The presentation prepared in METU Computer Engineering Department for the course CENG492 Spring 2011.

1.5. Document Overview

In introduction part, the purpose of the CMP, abbreviations and references are explained and stated. The organization of the team, responsibilities of team members and tools used during the development are described in the Organizations CM Framework part. Information about the identification process, tools and practices are described in the third section. Project Schedules and CM Milestones are described in the 4th section. The project resources that we would be using for CM is stated in Project Resources part. Finally in Plan optimization part we describe the methods to be followed for optimizing CMP.

2. Management

2.1. Organization

ONEV project is developed by iTeam4 group which is consisting of four computer engineers. The members of the group are:

- Hassan Salahe Matar
- Emilbek Joldoshev
- Hüseyin Lutin
- Mehmet Barış Özkan

Each of the group members focus on different aspect of the ONEV project.

2.1.1. Software Development Team

Targeted Members: ALL

SDT is implementing all modules in ONEV system. This team also integrates the implementations with the rest of the parts. Also the responsibilities of this team is creating releases, updating project source code via SVN and fixing the bugs.

2.1.2. Testing Team

Targeted Members: Mehmet *Barış Özkan, Hassan Salahe Matar*
This team is responsible of testing and debugging of the implementations produced by SDT. The team will prepare test cases and use cases to check whether the initial requirements are satisfied or not. According to test results the team gives feedback to SDT team.

2.1.3. Configuration Management Team

Targeted Members: Hüseyin Lutin, Mehmet Barış Özkan
Configuration Management Team is responsible for maintains of the CM organization. This team can update CM on necessary situations.

2.1.4. Configuration Control Team

Targeted Members: Emilbek Joldoshev, Hüseyin Lutin
The main responsibility of that this team is supervising all the activities of other groups. Also this group reviews SCRs, accepts or rejects SCRs and monitors the SCRs.

2.1.5. Release Control Team

Targeted Members: Emilbek Joldoshev, Hassan Salahe Matar
This team will control the current and next versions of the ONEV. Also it is this team responsibility to merge different branches of ONEV.

2.2. Responsibilities

Each member of ONEV is responsible from all the organizational teams as described above. Also it is all of team member responsibility to comment about the changes before committing resources through SVN.

2.3. Tools and Infrastructure

SVN (Subversion):

Apache Subversion (often abbreviated SVN, after the command name svn) is a software versioning and a revision control system founded and sponsored in 2000 by CollabNet Inc. Developers use Subversion to maintain current and historical versions of files such as source code, web pages, and documentation. Its goal is to be a mostly-compatible successor to the widely used Concurrent Versions System (CVS)[1].

SVN maintains the current and old versions of source codes, web pages and documentation on the server. We have configured RAD's SVN plug-in to commit and update our source codes.

Rational Application Developer:

IBM Rational Application Developer for WebSphere Software (RAD) is a commercial Eclipse-based integrated development environment (IDE), made by IBM's Rational Software division, for visually designing, constructing, testing, and deploying Web services, portals, and Java Enterprise Edition (JEE) applications[2].

We have used Rational Application Developer for development environment. Apart from the development environment duty it has some important tools that improve code quality, memory usage and threading problems.

TRAC:

Trac is an open source, web-based project management and bug-tracking tool. The program is inspired by CVSTrac, and was originally named svntrac *due to its ability to interface with Subversion.[1] It is developed and maintained by Edgewall Software*[3].

With the help of the TRAC, we will assign tasks to the members of SDT and also SCR's are raised. Every member of the team has to learn using Trac. Each member is going to use Trac in order to show his progress.

[1] http://en.wikipedia.org/wiki/Subversion_(software)

[2] http://en.wikipedia.org/wiki/IBM_Rational_Application_Developer

[3] http://en.wikipedia.org/wiki/Trac

3. Configuration Management Process

3.1. Identification

For configuration identification of ONEVS project, attributes that define every aspect of configuration items are identified. Configuration items of this project consist of 3 main parts; source codes, data, and documentation.

3.1.1. Source Code

Source code consists of three sub-items which are Graphical User Interfaces, Model Logic, and Control Logic

3.1.1.1. Election/Normal Mode Graphical User Interfaces

This consists of the following items;

- html templates,
- style sheets and
- images

These items are used in dynamic creation of web pages during interaction between clients like voters and the system.

3.1.1.2. Model Logic Classes

This consists of java jar files that define and implement the Database Persistence using Java Persistence API. They will facilitate interaction and querying of the control functions with the database which is DB2.

3.1.1.3. Control Logic Classes

This consists of the following items;

- ❖ Back-end service Java classes to realize various functions
- ❖ Administration Servlets and JSPs to dynamically create webpages according to clients' requests and the responses of back-end functional logic.

3.1.2. Database

For development purposes, every team member has DB2 database installed in his local machine. The common database definition is configured in the machines. The definition of the database (DDL) will be stored in SVN so that any member of the team can reconfigure their database, make changes, and commit the changes to the SVN. Additionally, dump file of data from the database, in sql script form, will be available in this item.

3.1.3. Documentation

This item contains the following documents;

- Project Reports
  - Project Proposal
  - Software Requirements Specifications Report
  - Initial Software Detailed Design Report
  - Final Software Detailed Design Report
  - Revised Detailed Design Report
  - Configuration Management Plan
- Weekly Progress Reports
- User Manuals
  - Installation Manual
  - How-To-Use Manual

3.2. Configuration Management and Control

3.2.1. System Change Requests

Minor System Change Requests are directly added to the system and handled by SVN and requires no extra information. However, major changes are controlled by the Trac system. In this system a System Change Requests should consist of

- Team member name
- Description
- Date
- Deadline
- Related Category/Module
- Priority, and

- Version number

When the request is made, a ticked is opened by Trac and it can be seen by all members of our team.

3.2.2. System Change Evaluation

The discussions about evaluation of System Change Requests are maintained on tickets in the Trac system. During the evaluation, each member write their opinions and the evaluation will take into account the opinions of the teams members.

3.2.3. System Change Implementation

If any System Change Request is approved after an evaluation, then the related Configuration Item will be updated, committed and upgraded to a new version through the SVN system.

3.3. Configuration Status Accounting

We have introduced configuration items in previous sections and the information regarding configuration items is needed to be stored because the control of configuration items gets complicated as more changes are made during development. Since keeping track of development process of the project is one of the most essential factors to be handled, we will use different ways to express these changes and updates at the same time. The mentioned information is made up of the following: configuration identifications, change request's information and information related to details of implementation. While project approaches to end, we will keep information about the details of configuration items in the comments of SVN commits and meeting reports are also a guide through the common changes. Moreover, versioning of the documents will be controlled via explaining clearly by comments.

3.4. Auditing

The team members will accomplish the task of auditing. During the auditing phase of system change requests, the changes that are made on a Configuration Item (CI) will be checked to determine their correctness. Auditing of the source code will be done during group meetings in Friday and Sundays by using appropriate testing methods. Auditing of data in the database definition will be done according to decisions made previously in the design phase. Moreover, every member should make sure that his code is working properly before each commit into the SVN repository. Each commit should at least be compiled correctly, or otherwise team member should state, in form of a comment, the source of the problem and the section in the code that is erroneous. By this policy, the source code in the repository is always trusted to be compiled and working or at least easy to correct the code. Besides these functional audits, members should observe the physical and process status each week. Project schedule should be checked and updated regularly in order to obey the timing constraints. Number of open tickets and resolved ones should also be evaluated in order to maintain sustainability.

4. Project Module And CM Milestones

The living schedule of iTeam4 that shows the tasks and milestones is already prepared and accessible from the ONEV project website. We are updating the living schedule weekly in terms of tasks which we are finished.

The milestones which the team follows can be listed as follows:

- First Development Snapshot Demo (15 April 2011)
- Pre-First Release Prototype (1 May 2011)
- First Release (15 May 2011)
- Final Release and Documentation (10 June 2011)

5. Project Resources

The resources of the project can be listed as follows:

- **SVN:** Revision Control System
- **Rational Application Developer :** Development Environment
- **TRAC :** Issue Tracking System
- **Web Site:** Project Development News

6. Plan Optimization

CMP will guide the team members on coordination and development of the ONEV. Since we have defined our tasks in the living schedule, all members of the team will follow these tasks. On the other hand, team may need to make some changes in the deadlines of the living schedule during the development process. When an update or change occurs, this will be followed by all group members via TRAC system. In the team's weekly meetings we decide the necessary changes and optimizations about the project and these changes are reflected to our living schedule which can be accessible from the ONEV website.