M.E.T.U

Computer Engineering Department
CENG 491

SOFTWARE CONFIGURATION
MANAGEMENT PLAN

Group Name: Babylon

Group Members:
  M. Tuğberk İşyapar
  Arif S. Özçelik
  Okan T. Tursun
  Ö. Anıl Türkkan

26, March 2010
CONTENTS

1 . Introduction 3
   1.1 . Purpose of this Plan 3
   1.2 . Scope of this Plan 3
   1.3 . Key Terms and Definitions 3
   1.4 . References and Resources Used 4

2 . The CM Framework Organization 4
   2.1 . Organization 4
      2.1.1 . Software Development Group 5
      2.1.2 . Hardware Supplement Group 6
      2.1.3 . Testing Group 6
      2.1.4 . Configuration Control Group 6
      2.1.5 . Release Control and Maintenance Group 7
   2.3 . Resources and Environment 7

3 . The CM Process 8
   3.1 . Identification 8
      3.1.1 . HammurAbi Source Code 8
      3.1.2 . Documentation 9
   3.2 Configuration Management and Control 9
      3.2.1 . Change Request 9
      3.2.2 . Evaluation 10
      3.2.3 . Finalizing Requests 10
   3.3 . Configuration Status Accounting 11
   3.4 . Auditing 11

4 . Project Schedule (GANTT Chart) & Milestones 12

5 . Project Sources 13

6 . Plan Optimization 13
1. Introduction

1.1. Purpose of this Plan

The purpose of this document is to define a configuration management process for our senior year project, HammurAbi. As a team we aim to develop a Vehicular Ad Hoc Network software. During the development of such a system, the team members will be changing and editing many things related to the source code of the project, as they progress towards finalizing it. Thus there should be a clear definition of a controllable environment in which we can maintain both the stability and integrity of the software to be developed.

This plan is prepared so that the problems discussed above can be handled within a well defined frame.

1.2. Scope of this Plan

This report describes the standards of our development process and defines certain responsibilities that will be assigned to team members. It also gives the reader information about the schedule, milestones and certain other crucial points regarding the CM.

1.3. Key Terms and Definitions

- CM – Configuration Management
- CMP – Configuration Management Plan
- CI – Configuration Item
• CCB – Configuration Control Board
• CSA – Configuration Status Accounting
• JVM – Java Virtual Machine
• PDA – Personal Digital Assistant
• SVN – Subversion
• VANET – Vehicular Ad Hoc Network
• API – Application Programming Interface

1.4. References and Resources Used

While preparing this plan, we made use of
• The slides from the presentation made on “Software Configuration Management Plans” during class of Ceng492, a senior course given by the department of Computer Engineering.
• The documents and reports previously published by team Babylon, on their project HammurAbi. Examples are Software Requirements Analysis Report, Initial Design Report and Final Design Report.

2. The CM Framework Organization

2.1. Organization

The team consists of 4 members:
• Okan T. Tarhan
Tasks are assigned to each member on a case by case approach. Currently, the team work is concentrated in two main areas. The first one is developing the functionalities of the software and binding a GUI with them. The other concentration is to develop a communication between the main component of HammurAbi and the simulator GrooveNet that supplies simulated data for each vehicle in the system, so that a real time VANET can be simulated.

As the development process progresses and each task is accomplished, the team expects to concentrate on other aspects related to the project. Thus assigning work to team members is a highly dynamic matter. Further details on other duties on CM can be found in the following sections.

2.1.1. Software Development Group

This group is in charge of developing the necessary source code.

Every member of the team is responsible of developing the software. Team members may be assigned to concentrate on certain areas from time to time.
2.1.2 . Hardware Supplement Group

This group is in charge of supplying the hardware platform needed to run the software. The project does not include developing any kind of hardware, so this group basically finds a suitable platform that satisfies certain requirements. For example, HammurAbi is intended to be used in vehicles with touchpad screens that are also suitably sized to be used in such environments. Therefore a feasible platform is needed.

Team members Okan and Arif are assigned to this group.

2.1.3 . Testing Group

This group is in charge of testing and debugging the code.

Team members: Anil and Arif are assigned to this group.

2.1.4 . Configuration Control Board

This group will be monitoring the SCM process the whole time. It will inform the team if there are inconsistencies after any system changes or updates and will make decisions in such cases. It will also update the CM schedule and coordinate the team members.

All members take part in this group. As the team is small, this makes it unnecessary to centralize the decision making mechanism.
2.1.5 . Release Control and Maintenance Group

This group will be in charge of planning and controlling future versions of the software. They will also be responsible for maintaining releases.

Team members: Okan and Tugberk

2.3 . Resources and Environment

For Software Development & Final Release:

- The source – code is being developed in JAVA.
- The environment the software will be running in will most probably be a PDA that has a JVM running on it.
- There will be Linux operating system running on the PDA.
- Eclipse IDE for coding.

For Software Maintenance:

Version control is the management of changes to documents and other information stored as computer files. It is used to indicate any inconsistencies related to these documents and keep the integrity of the information stored.

SVN is a version control system that is used to maintain current and historical versions of files such as source – code, web pages and other similar documents. This tool stores the central information in what is called a “repository” or “trunk”. Users can make branches of or in technical terms “check out” this pool of information, i.e. download a copy of repository and edit or update this copy. Then the modified copy
of the repository is “commited” back to the version control system to be handled by SVN.

3. The CM Process

3.1. Identification

We have two CIs that can be defined in our process,

- Source Code
- Documentation

3.1.1. HammurAbi Source Code

Our source code is written in Java on the Eclipse platform. We will be further developing our code in an organized fashion, using the concepts of version control systems. This is very important because we will be continually modifying our code.

While developing the code, we pay attention to write it to be modular. This helps us organize and concentrate our work in specific areas without worrying about any incoordination. Also we will try to emphasize more comments in our code for a better communication between team members. The CCB will be responsible for auditing these rules during the CM process.
3.1.2 . Documentation

The documentation related to a software project is important because it “documents” the past of the project. It connects its past to its current date. The documents so far produced by our team are:

- Project Proposal
- Software Requirements Analysis Report
- Initial Design Report
- Final Design Report
- Revised Design Report
- Web site that we will publish every other document in the end
- Configuration Management Plan
- Weekly Reports

There are two more documents that will be published in the future:

- Test Specifications Report
- User Manual

3.2 . Configuration Management and Control

3.2.1 . Change Request

Request are made by team members in case they want to change/modify/update part of a configuration item. The requests may be delivered via mail, using our mail group but most of the time the software TRAC which is compatible with SVN will be used to notify the team about request. In a TRAC request ticket the following details should be provided by requester:
• Date of request
• Version to apply the request
• Components or modules the request is about
• Deadline of the request
• Detailed description of the request
• Urgency of the request
• Type of request (defect, enhancement, task)
• Owner of the request
• The assigned team member who if accepted will be applying the request

3.2.2 . Evaluation

Auditing and evaluating received request tickets is up to the CCB. But we are a decentralized and small team. So our CCB basically is formed by the whole team. The mail groups and weekly meetings are convenient locations to discuss incoming requests.

It is important to the team and project that each member utilizes his time efficiently and checks the TRAC e-mail group as often as possible. Additionally, a key feature is to establish good communication among team members.

3.2.3 . Finalizing Requests

After the CCB discusses the request and makes a cost/profit analysis, it decides on whether to implement the necessary
modifications/updates. If it agrees with the requester, a team member is assigned to perform the task. This assignment may be the one suggested by the requester, but does not have to be. In certain cases a better suited candidate may be assigned to the same task.

3.3. Configuration Status Accounting

CSA is the sum of activities of recording and reporting recent and crucial developments on CIs. Weekly meetings and weekly reports are tools that are used to inform the course instructors and assistants on such matters. In addition to these meetings, SVN provides the team members a tool to record every modification/update that has been made on the source code. The web site may also be used to report any progress or evolvement. Lastly, the comments written by each team member while developing the code is very important, as was mentioned before. They also are a good source of recordings of recently made changes. They tend to be more descriptive in details and thus more informative.

There will be a text file stored at the team’s web site. Each member shall add what changes they make on the source code together with the date of the change to this text file.

3.4. Auditing

After every weekly meeting the team will discuss their progress so far, the general state of the project and criticism and comments made by the instructors and/or assistants during the meeting or demos. Besides this the mailing group will be the most frequently used tool for this purpose.
### Project Schedule (GANTT Chart) & Milestones

<table>
<thead>
<tr>
<th>ID</th>
<th>Task Name</th>
<th>Start</th>
<th>Finish</th>
<th>Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GUI Implementation</td>
<td>22/02/2010</td>
<td>16/04/2010</td>
<td>8w</td>
</tr>
<tr>
<td>2</td>
<td>Implementation of main window</td>
<td>22/02/2010</td>
<td>05/03/2010</td>
<td>2w</td>
</tr>
<tr>
<td>3</td>
<td>Implementation of zooming and painting functions</td>
<td>08/03/2010</td>
<td>19/03/2010</td>
<td>2w</td>
</tr>
<tr>
<td>4</td>
<td>Implementation of mouse functions and padding</td>
<td>22/03/2010</td>
<td>08/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>5</td>
<td>Implementation of menus and dialog windows</td>
<td>28/03/2010</td>
<td>16/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>6</td>
<td>Modifying Simulator Source Code</td>
<td>12/03/2010</td>
<td>01/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>7</td>
<td>Setting up network communication and message forwarding</td>
<td>12/03/2010</td>
<td>01/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>8</td>
<td>Network Communication</td>
<td>22/02/2010</td>
<td>16/04/2010</td>
<td>8w</td>
</tr>
<tr>
<td>9</td>
<td>Implementation of message format</td>
<td>22/02/2010</td>
<td>05/03/2010</td>
<td>2w</td>
</tr>
<tr>
<td>10</td>
<td>Implementation of network sockets</td>
<td>08/03/2010</td>
<td>28/03/2010</td>
<td>3w</td>
</tr>
<tr>
<td>11</td>
<td>Parsing network messages and creating entities on map</td>
<td>29/03/2010</td>
<td>16/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>12</td>
<td>Milestone: First Development Snapshot</td>
<td>06/04/2010</td>
<td>06/04/2010</td>
<td>0w</td>
</tr>
<tr>
<td>13</td>
<td>Map functions</td>
<td>22/03/2010</td>
<td>30/04/2010</td>
<td>6w</td>
</tr>
<tr>
<td>14</td>
<td>Implementation of map entity representation</td>
<td>22/03/2010</td>
<td>09/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>15</td>
<td>Implementation of layers</td>
<td>29/03/2010</td>
<td>16/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>16</td>
<td>Row finding</td>
<td>12/04/2010</td>
<td>30/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>17</td>
<td>Core System Functions</td>
<td>20/03/2010</td>
<td>23/04/2010</td>
<td>4w</td>
</tr>
<tr>
<td>18</td>
<td>Monitoring status of own vehicle</td>
<td>29/03/2010</td>
<td>09/04/2010</td>
<td>2w</td>
</tr>
<tr>
<td>19</td>
<td>Reading/writing configuration files</td>
<td>05/04/2010</td>
<td>23/04/2010</td>
<td>3w</td>
</tr>
<tr>
<td>20</td>
<td>Milestone: First Release Demo</td>
<td>13/05/2010</td>
<td>13/05/2010</td>
<td>0w</td>
</tr>
<tr>
<td>21</td>
<td>Testing and debugging</td>
<td>03/05/2010</td>
<td>14/05/2010</td>
<td>2w</td>
</tr>
<tr>
<td>22</td>
<td>User Documentation</td>
<td>10/05/2010</td>
<td>21/05/2010</td>
<td>2w</td>
</tr>
<tr>
<td>23</td>
<td>Packaging and release</td>
<td>17/05/2010</td>
<td>28/05/2010</td>
<td>2w</td>
</tr>
<tr>
<td>24</td>
<td>Milestone: Final Package Submission</td>
<td>10/06/2010</td>
<td>10/06/2010</td>
<td>0w</td>
</tr>
</tbody>
</table>
5. Project Sources

The sources comprise of those mentioned in section 2.3 (Resources and Environment – Software Development & Final Release) and section 3.1.2 (Documents) and the documentation of the OpenMap API.

6. Plan Optimization

This plan is theoretical framework to implement our CM. We are aware that of course unsuspected changes are part of our practicality. So modification of the CM is possible. In such a case the team will discuss the situation and decide on a way to optimize the plan.