MIDDLE EAST
TECHNICAL UNIVERSITY

COMPUTER ENGINEERING
DEPARTMENT

CENG 492
CONFIGURATION MANAGEMENT PLAN
WEBMES

TOMBEKI++
TEAM NAME : TOMBEKI++
PROJECT : WEBMES
ASSISTANT : CAGATAY CALLI
SUPPORTER COMPANY : AGMLAB

PREPARED BY:
M.ARIF UGUREL – 1348036
BUGRA HASBEK – 1347541
RUSLAN ABDULLA – 1321751
SERDAR SAYGILI – 1347897
TABLE OF CONTENTS

1 INTRODUCTION.................................................................................................................. 4
  1.1 THE PURPOSE OF CONFIGURATION PLAN ................................................................. 4
  1.2 THE SCOPE OF CONFIGURATION PLAN .................................................................. 4
  1.3 DEFINITIONS, ACRONYMS AND ABBREVIATIONS .................................................. 5

2 CM FRAMEWORK ORGANIZATION .............................................................................. 6
  2.1 THE ORGANIZATION OF SCM TEAM .................................................................... 6
  2.2 THE RESPONSIBILITIES OF SCM TEAM ................................................................. 7
  2.3 TOOLS AND INFRASTRUCTURE ............................................................................ 8

3 CONFIGURATION MANAGEMENT PROCESS ............................................................ 9
  3.1 IDENTIFICATION ....................................................................................................... 9
    3.1.1 JAVA SOURCE CODE ....................................................................................... 9
    3.1.2 DATA ............................................................................................................... 9
    3.1.3 DOCUMENTS .................................................................................................. 9
    3.1.4 WEB PAGES ................................................................................................ 10
  3.2 CONFIGURATION MANAGEMENT AND CONTROL .................................................. 10
    3.2.1 REQUESTING CHANGES ............................................................................... 10
    3.2.2 EVALUATING CHANGES ............................................................................... 10
    3.2.3 IMPLEMENTING CHANGES ............................................................................. 11
  3.3 CONFIGURATION STATUS ACCOUNTING ............................................................... 11
  3.4 AUDITING .............................................................................................................. 11

5 PROJECT SCHEDULES & CM MILESTONES .............................................................. 12

5 PROJECT RESOURCES..................................................................................................... 13

6 PLAN MAINTENANCE ..................................................................................................... 13
1 INTRODUCTION

1.1 The Purpose of the Configuration Management Plan

Nowadays it is very hard to find a well-defined project in IT market. In majority of software projects, the job is subject to change at almost any moment. If a company cannot cope with customer’s request for changing some parts of software then the company will lose not only money but also reputation of company in the market. If a project is well-designed then the company will have a big advantage. However this advantage can easily disappear if the company is not ready for changes in problem definition.

The main handicap of changes in project is, it can have catastrophic effects on a seemingly irrelevant part of project. Uncoordinated teams tend to do changes in project with insufficient evaluation and documentation. Such teams are destined to doom in the middle of a project because of the unavailability of configuration management plan.

Tombeki team’s main goal for writing this document is preserving the integrity of Marpuch project until it finishes by defining our policies for version controlling. When we think to do a change in Marpuch; we will decide whether that change is needed, how it will be integrated and how it will be documented in the light of this document.

1.2 The Scope of the Configuration Management Plan

Software configuration management activities of Marpuch are the main scope of this document. One of our top priority goals is having a maintainable and flexible end product. So we give this document great importance and we are planning to obey procedures defined in this document throughout the semester.

Our configuration management teams and their respective responsibilities are also declared in this document. We believe that this team structure will prevent responsibility confusions in the coming months.

SCM tools which we will use for Marpuch project is also in the scope of this document. CM milestones will also be covered in this document.
### 1.3 Definitions, Acronyms and Abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CM</td>
<td>Configuration management</td>
</tr>
<tr>
<td>SCM</td>
<td>Software configuration management</td>
</tr>
<tr>
<td>SCMP</td>
<td>Software configuration management plan</td>
</tr>
<tr>
<td>CT</td>
<td>Configuration teams</td>
</tr>
<tr>
<td>CMT</td>
<td>Configuration management tools</td>
</tr>
<tr>
<td>FTR</td>
<td>Formal technical review</td>
</tr>
<tr>
<td>CI</td>
<td>Configuration item</td>
</tr>
<tr>
<td>SDT</td>
<td>Software development team</td>
</tr>
<tr>
<td>TT</td>
<td>Testing team</td>
</tr>
<tr>
<td>CCT</td>
<td>Change control team</td>
</tr>
<tr>
<td>VCT</td>
<td>Version control team</td>
</tr>
<tr>
<td>RCT</td>
<td>Release control team</td>
</tr>
<tr>
<td>CMUT</td>
<td>Configuration management updates team</td>
</tr>
</tbody>
</table>
2 CM FRAMEWORK ORGANIZATION

2.1 The Organization of SCM Teams

1. Software Development Team (SDT):
   a. Bugra HASBEK
   b. Arif UGUREL
   c. Ruslan ABDULLA
   d. Serdar SAYGILI

2. Testing Team (TT):
   a. Bugra HASBEK
   b. Ruslan ABDULLA

3. Change Control Team (CCT):
   a. Arif UGUREL
   b. Bugra HASBEK

4. Version Control Team (VCT):
   a. Arif UGUREL

5. Release Control Team (RCT):
   a. Bugra HASBEK
   b. Ruslan ABDULLA

6. Configuration Management Update Team (CMUT):
   a. Arif UGUREL
   b. Serdar SAYGILI
2.2 The Responsibilities of SCM Teams

Tombeki team has 4 members and each of them is part of a SCM team. Responsibilities of teams are:

1. Software Development Team: This team’s main task is implementing Marpuch modules. Moreover SDT is responsible for creating releases, updating project source code via SVN, fixing bugs.

2. Testing Team (TT): This team’s main task is finding bugs in Marpuch. Moreover this team will create test cases and use cases, testing whether initial requirements are satisfied or not.

3. Change Control Team (CCT): This team is responsible for the administration of other teams. CM activities will be accepted or rejected by CCT.

4. Version Control Team (VCT): This team will monitor versions of Marpuch. Moreover VCT is responsible for merging different branches of Marpuch

5. Release Control Team (RCT): This team will monitor the progress of releases.

6. Configuration Management Update Team (CMUT): This team is responsible for updating configuration management plan and living schedule.

Tombeki members also have common CM responsibilities in addition to upper responsibilities. These responsibilities are:

1. Notifying all members about changes on source codes via mobile phone.

2. Following updated CM schedule and obeying deadlines.

3. Informing other team members about his progress.

4. Obeying team conventions about commenting and code writing.

5. Committing written code regularly.
2.3 Tools and Infrastructure

We will use Subversion (SVN) for version controlling of Marpuch. SVN is a widely used version controlling software. SVN has some very handy features which tempted us to it as our main SCM tool. These features are:

- Commits are true atomic operations. Interrupted commit operations do not cause repository inconsistency or corruption.
- Renamed/copied/moved/removed files retain full revision history.
- Directories, renames, and file metadata (but not timestamps) are versioned. Entire directory trees can be moved around and/or copied very quickly, and retain full revision history.
- Branching and tagging are cheap operations, independent of file size.
- Client/server protocol sends diffs in both directions.
- Costs are proportional to change size, not data size.
- Open source licensed — "CollabNet/Tigris.org Apache-style license"
- Internationalized program messages.

We use netbeans as our programming ide. Interaction between netbeans and SVN happens in an indirect way. When a group member adds a new file to repository, other group members update their project directory. Newly added file appears in netbeans project because it is in the project directory.

Since netbeans’ project directory structure is quite practical, we are not planning to have a different directory structure. So we will not artificially separate files with respect to their types under a root directory. However there will still be the order of netbeans, in the repository directory. For example following files will be in the same directory:

- Images
- Java source codes
- JSP source codes
3 CONFIGURATION MANAGEMENT PROCESS

3.1 Identification

3.1.1 Java Source Code
Our source code will be subject to change throughout the semester. Marpuch is a modularly designed project so keeping the track of changes in Marpuch will be relatively easy even though it is of utmost importance. Java source codes will be put under the same directory for easy access.

3.1.2 Data
Background images, developer photographs, rating system icons and user profile images is our data configuration items. All of this data will reside in the same directory. We have relatively less data item so putting them all in the same directory will not cause confusions.

3.1.3 Documents
We believe that documenting a project is as important as writing source codes. So we pay attention for documentation of Marpuch. Some of the documents that we have already created and published in Tombeki web page are:

- Project proposal
- Requirements analysis report
- Initial design report
- Final design report
- Living schedule
- Configuration management plan

There are also some documents that we are planning to publish in the coming months. These documents are:

- Test cases for alpha testing and unit testing
- User documentation of Marpuch

Our living schedule is going to be dynamic. So Ceng 492 stuff and our respectful competitors will be able to see our progress.
3.1.4 Web Pages

Web pages will be put in the same directory however it is possible that some other stuff may also reside in that directory. We think that it is easy enough to find the web page that we seek so we did not attempt to create some kind of cryptic directory structure.

3.2 Configuration Management and Control

3.2.1 Requesting Changes

If a team member believes that a part of project should be changed then he should:

- Take notes of the code portion which is subject to change request.
- Take notes of the reasons of change request
- Send a softcopy of his notes to change control team

3.2.2 Evaluating Changes

Change control team members will make a meeting each Thursday night and if a change request has been done by a group member then cct will evaluate request according to following principles:

- Change requester will be in the meeting.
- Change requester will explain his reasons by using notes on the softcopy of change request.
- Both CCT members should vote for a change, otherwise the change request will be rejected.
3.2.3 Implementing Changes
If the request is accepted than the procedure to follow is:

- If there is an above average chance that the change will have undesirable side effects then an experimental branch will be created in SVN.
- After every change in the project, softcopy of change request will be attached to the comment part commit.
- After extensive unit testing, experimental branches will be merged with the main stream.

3.3 Configuration Status Accounting
CSA is an important part of CM because it is easy to get lost in big projects. So we will prepare large and descriptive comments when we make a commit to repository. If the commit is done due to a change request then we will add the softcopy of change request to comment part. Moreover softcopies of change requests will reside in our mailboxes so that we will easily look at softcopies to recall why we made a certain change in project.

3.4 Auditing
Auditing is vital for the success of a project. People tend to do unconscious auditing almost everyday. Coffee machines are a good place for such little auditings. We believe in the power of such little auditings. We work on Marpuch generally in the same project and whenever an obstacle prevents us from going further, we make little audits. Moreover every week we make a big audit on Thursday where we discuss the project in general.
4 PROJECT SCHEDULES and CM MILESTONES

Having short term goals is a good way increasing efficiency of team. Due dates are infamous for creating too much stress in software developers however we believe that it has positive effects on us. For these reasons we set some milestones for project Marpuch.

Our milestones are based on finish of modules and tests. Here are are milestones:

- Implementation of Displayer Module : 19.03.08
- Implementation of Authentication Module : 26.03.08
- Implementation of Analyzer Module : 20.04.08
- First Alpha Release : 27.03.08
- Implementation of Machine Learning Module : 20.05.08
- Integration of Modules : 25.05.08
- Integration Test : 27.05.08
- Second Alpha Release : 27.05.08
- Alpha Testing : 30.05.08
- Integration and Documentation : 15.06.08
5 PROJECT RESOURCES

We have 3 main CM resources:

1. Tombeki team members  every member is part of one of the CM teams.
2. SVN  our main version controlling software. We will use svn extensively.
3. Marpuch Web page  we will put living schedule, screenshots and project documentation on our web page.

6 PLAN MAINTENANCE

We may revise this document in the coming months according to our needs. Configuration Management Update Team is responsible for updating this document.