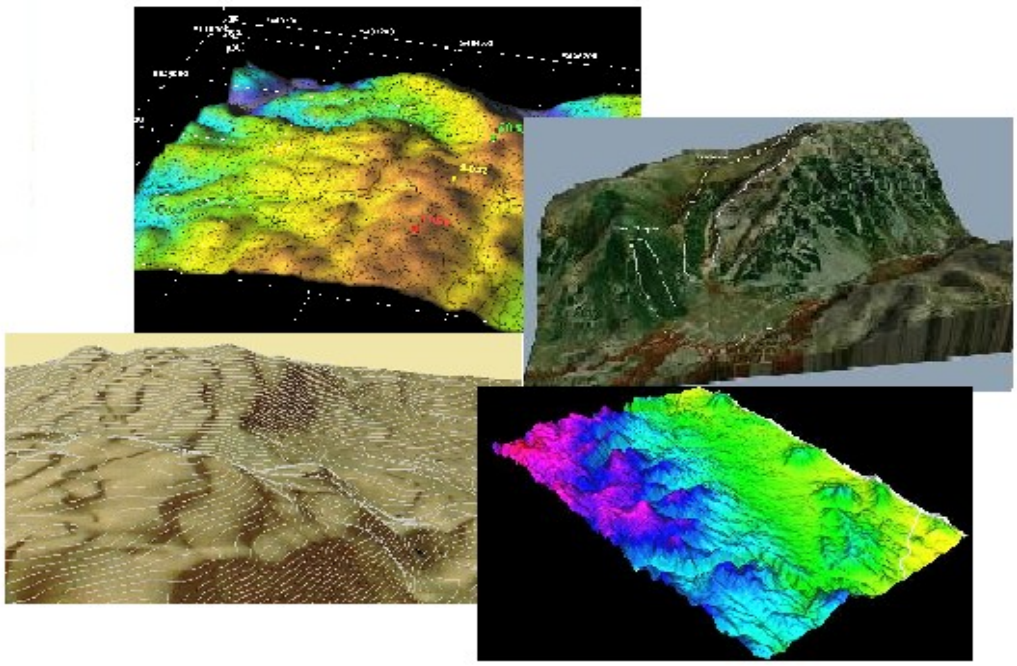
Sirius Software

Duygu Yapa 1449222

Ayşe Turan 1449164

Duygu Altınok 1502095

Elif Kamer Karataş 1448836



THE STRIDER

[Configuration Management Plan]

09.03.2009

1. INTRODUCTION....3
1.1 Purpose of CMP....3
1.2. Scope of the Document....3
1.3 Definitions, Acronyms and Abbreviations....4
2. CM FRAMEWORK ORGANIZATION....4
2.1. Organization....4
2.2. Responsibilities....5
2.3. Tools....5
3. CONFIGURATION MANAGEMENT PROCESS....6
3.1. Identification....6
3.1.1. Source Code CI....6
3.1.2. Data CI....6
3.1.3. Document CI....7
3.2. Management and Control....7
3.2.1. Change Requests....8
3.2.2. Change Request Report....8
3.2.3. Change Request Evaluation....8
3.2.4. Change Implementation....8
3.2.5. Defect Tracking....9
3.2.6. Development Management and Control....9
3.3. Configuration Status Accounting....9
3.4. Auditing....9
4. PROJECT SCHEDULES AND CM MILESTONES....10
5. PROJECT RESOURCES....11
6. PLAN OPTIMIZATION....11

1. INTRODUCTION

1.1 Purpose of CMP

In software world, most of the projects are long term projects. During the life time of a project the customers most probably want some extra features even they have no idea about feasibility of their requests. If the customer is also an information technology firm, you are lucky because they know what they want and they can prepare their requirement reports well and clearly. But if the customer is a government agency or some other firms, you have too much work. Generally they can not specify their needs well initially, because they don't much know about software. So, they generally change their requests or add some extra to the project scope during the project development. Hence, as a software company, we should handle the changes which take root from customer or even us, under configuration management title.

On the other hand, this project is our first serious software project so some of our previsions may not be realistic and they can be changed.

1.2. Scope of the Document

In this document, we mention about how the configuration management process in our project is handled and what the responsibilities of the group members on this issue are.
CM Framework Organization and the responsibilities, tools and infrastructure are given in the second section. How our CM process will be handled is given in the third section. In the 4. section of this document, you can see our CM milestones. Project resources and plan optimization are also in the scope of this document.

We are aiming at having a maintainable and flexible end product and we are planning to obey the rules we defined in this document.

1.3 Definitions, Acronyms and Abbreviations

CM Configuration Management

CMP Comfiguration Management Plan

SCM Software Configuration Management

CCB Configuration Control Board

SDT Software Development Team

TT Testing Team

CMUT Configuration Management Updates Team

CI Configuration Item

2. CM FRAMEWORK ORGANIZATION

2.1. Organization

In order to carry out SCM activities we've composed 4 teams. The name of these teams and their members are as follows.

- **Configuration Control Board (CCB):** Sirius Software is a small company with 4 members, so all company members are also the members of the Configuration Control Board.
- **Software Development Team (SDT):** All members of the Sirius Software are also the members of this team.
- **Testing and Debugging Team (TT):** Ayşe TURAN-Elif Kamer KARATAŞ
- **Configuration Management Update Team (CMUT):** Duygu YAPA-Duygu ALTINOK

The responsibilities assigned for the above teams will be given in the following section.

2.2. Responsibilities

CCB: This team is responsible for the change control of configuration items. In this context it coordinates the activities of the other 3 teams. It evaluates the changes requests coming from either Software Development Team or Testing & Debugging Team and decides whether performing the change is reasonable or not.

SDT: The main responsibility of this team is to implement the modules of the "Strider Project". Team members are obligated to use SVN and update the project repository during the development process. This team can make change requests when necessary. It also implements the required changes according to the feedback coming from the Testing &Debugging team.

TT: The responsibility of this team is to test the functionality of the modules implemented by SDT. It searches for the possible bugs. It checks if the requirements are satisfied. In case of any errors or bugs this team is responsible to make the related change request to CCB.

CMUT: This team is responsible for the update of CMP and living schedule of the project.

2.3. Tools

SVN: We will use SVN for CMP and CM activities. SVN is an open source Version Control System. It manages files and directories over time. Files are placed into a central repository and any change ever made to the files or directories are remembered.

NetBeans: We decided to use NetBeans platform for development instead of Eclipse because of it's more easy to design and change graphical user interfaces.

3. CONFIGURATION MANAGEMENT PROCESS

3.1. Identification

When designing Strider, we have divided the project into distinct components that can be developed independently. It is a modularly designed project and it already consists of independent processes. For this reason, it is quite simple to identify the configuration items. There are 3 main configuration items; namely Source Code CI, Data CI and Document CI.

3.1.1. Source Code CI

Source Code CI consists of all the implemented modules of the Strider project. We divide the project into packages. Packages consist of classes and classes consist of their methods. Declarations about classes and methods are commented for understandibility and reusability. All package, class and method names are given meaningfully. All packages are located under “src” directory.

Here are the subdirectories under the “src” directory:

definedTypes	This directory contains definedTypes package.
gis	This directory contains gis package.
gui	This directory contains gui package.
planner	This directory contains planner package.

3.1.2. Data CI

Data CI consists of all the data items used by the program.

Here are the subdirectories under “data” directory:

images	This directory contains images used in GUI.

tables	This directory contains database tables namely Member and Equipment and the SQL scripts for creating those tables.
maps	This directory contains raster and vector map samples.

3.1.3. Document CI

Documentation is one of the most important part of a project. It must be clear and understandable.

Here are the documents that we have already written:

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

Here are the documents that we will write:

- Test Specifications
- User Documentation

All of the documents listed above will be included as document CIs. Living Schedule will be updated regularly and the documents can be retrieved from our website.

3.2. Management and Control

We have divided our work into parts as independent as possible. The development of each part is under different person's responsibility. (Duygu Yapa deals with GIS, Ayşe Turan deals with GUI, Elif Karataş and Duygu Altınok deals with algorithms)

3.2.1. Change Requests

During the development of the Strider any member of the team can request any change any time. Since each person deals with a different portion, change request reports should be prepared only if the change is major or it can affect others work. Minor changes can be stated informally; change request report is not necessary for minor changes. But, the purpose of the change and the change should be stated.

3.2.2. Change Request Report

Here is the format of the change request report:

- Name of the member making the request
- Date of the request
- File name of the source code
- Reason for change
- Proposed changes

3.2.3. Change Request Evaluation

In weekly meetings, change requests belonging to the previous week are discussed. The reasons fort he changes, their priorities and effects is taken into account during the evaluation. Since the member developing the part of the code which is wanted to be changed will make the desired change, her opinion will be taken especially.

3.2.4. Change Implementation

If the change request is approved, the member developing the part of the code which is subject to change will make the necessary changes.

3.2.5. Defect Tracking

All members are responsible for defect tracking for a reliable software. They should report all defects detected in the software. For removing defects, change request should be done and evaluated to determine the responsible person to eliminate the defect.

3.2.6. Development Management and Control

We use Java Coding Conventions in the development of our project for understandibility. And every class has declarations including

- name of the file
- name of the contributors
- creation date
- description of the class
- change history (including the person who made the change, date of the change, reasons to change and changes made)

3.3. Configuration Status Accounting

Since this is a relatively small project compared to other commercial projects, and we are 4 people sharing the code, we think that it's enough to use our mail group to state our current status, changes and bugs. The important thing here is that the explanations should be made for every change in case of the risk for forgetting it. Also, the decisions and bugs should be documented with reasons.

3.4. Auditing

Auditing will be performed on the basis of requirements. Before snapshots and releases we will check the closeness of us to the requirements that we have decided before. Necessary tests will be performed for a reliable software. Also, fort he change requests and commits auditing will be performed for determinin whether the desired functionality is accomplished or not.

4. PROJECT SCHEDULES AND CM MILESTONES

Because of the lack of time and source (four member project), configurations are managed by person who is responsible for the configured part. Configurations will be reported at schedule and all members can see that who change which part of project. Also the configurations should be accepted by other members. When the first alpha version is completed, members will focus other parts and if they have difrerent ideas or improvments, these are debated by group members. At the end, all group members have rights about configuration of the project.

We'll inform our assistant about our progress in weekly meetings and consider the given feedbacks. We will make short demos in every 2 weeks after the first snapshot demo. We'll continue to hold our own weekly meetings as a group. We've prepared a Living Schedule for our development process. As we complete the assigned tasks we'll update the living schedule.

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

First Compilable Version : 12.03.2009
Implementation of GIS Modules : 15.03.2009
Implementation of GUI Modules : 15.03.2009
Implementation of Algorithms : 30.04.2009
Integration of GIS and GUI : 19.04.2009
Integration of GIS, GUI and Algorithms : 03.05.2009
First Release : 03.05.2009
Testing & Bug Elimination : 17.05.2009
Second Release : 25.05.2009
Testing & Bug Elimination : 30.05.2009
Documentation (Preparing User Manual) : 26.04.2009

Final Release : 07.06.2009

5. PROJECT RESOURCES

For CMP and CM activities we will use SVN. And, we will put our documents and living Schedule, screen shots in our website.

6. PLAN OPTIMIZATION

Although we will do our best to obey the schedule and this plan, there may be slight changes on the dates and orders of the processes. Since we meet regularly, and we can communicate through emails, the flow of information is fast but optimization is necessary to keep with the schedule.