MIDDLE EAST TECHNICAL UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING

ServerTheon Project Configuration Management Plan

BY

TheonTech

Table of Contents

1 Introduction

1.1 Purpose of Software Configuration Management Plan

Software configuration management (SCM) is a set of activities designed to control change by identifying the work products that are likely to change, establishing relationships among them, defining mechanisms for managing different versions of these work products, controlling the changes imposed, and auditing and reporting on the changes made. [1]

The purpose of the ServerTheon SCM Plan is to ensure the products generated by TheonTech are adequately stored, managed; changes to the products are controlled; change processing and implementation status are recorded and reported; and compliance with specific requirements is verified.

1.2 Scope of Document

This document constitutes Software Configuration Management Plan of TheonTech for the project named ServerTheon. The plan applies to whole process of ServerTheon development, and SCM process described in this plan will be applied to all configuration items which are also identified in this document. This plan constructs a basis for identification, documentation, implementation, verification, audition and approval of changes throughout the lifecycle of ServerTheon project.

[1] Pressman, R. S. 1997, Software Engineering: a Practitioner's approach, 4th ed., New York: McGraw-Hill Companies.

1.3 Definitions, Acronyms and Abbreviations

ACRONYMS	DEFINITIONS
CI	Configuration Item
CM	Configuration Management
CMC	Configuration Management Committee
CMP	Configuration Management Process
CSA	Configuration Status Accounting
CVS	Concurrent Versioning System
FCA	Functional Configuration Audits
SCM	Software Configuration Management
SCR	System Change Request

1.4 Document References

- IEEE Std 828-1998, IEEE Standard for Software Configuration Management Plans

2 CM Framework

2.1 Organization

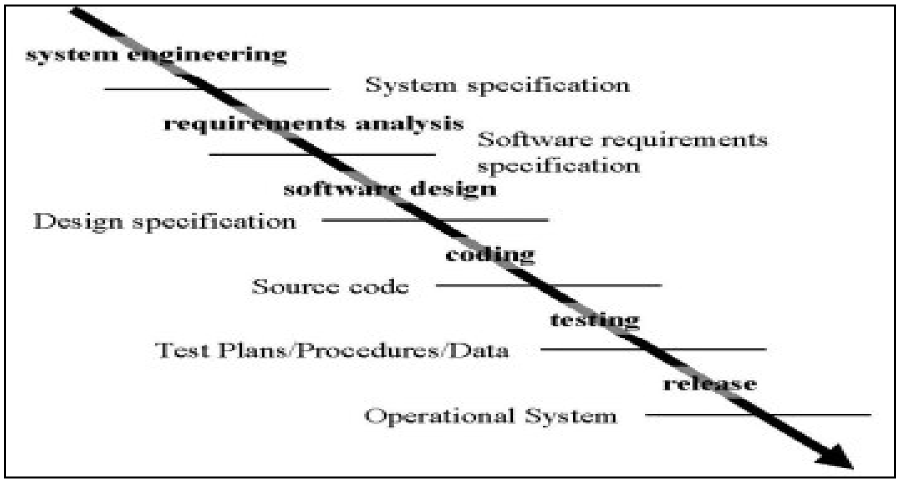
Since TheonTech consists of four members, we have preferred to have only one organizational unit. TheonTech have founded a Configuration Management Committee (CMC) to implement the planned SCM activities. According to the IEEE's (IEEE Std. 828-1990) traditional definition of SCM, these SCM activities are configuration identification, configuration control, configuration status accounting (CSA) and functional configuration audits (FCAs).

2.2 Responsibilities

We have already identified the SCM activities in the previous part. Now, we will define the responsibilities of CMC to implement each SCM activity.

To implement configuration identification, the responsibilities of CMC are:

- Dividing the system into uniquely identifiable components, Configuration Items (CIs)
- Identification of the project's baselines and their contents (see Figure-1)



(Figure-1)

To implement configuration control, the responsibilities of CMC are as follows:

- Evaluate and approve or disapprove System Change Requests (SCRs)
- Assign proposed changes to developers for implementation
- Verify changes in various testing procedures

To implement CSA and FCA, the responsibilities of CMC are as follows:

- Recording and reporting of information needed to manage a configuration
- Ensure that the software product has been built according to FCA

2.3 Tools & Infrastructure

For version controlling purposes, all the source codes are stored on the central CVS server. Since CVS keeps track of all changes in source code files or documents, it allows our team to work on same source codes collaboratively. Since CVS uses client-server architecture, our team will be able to work on implementation of a class or a module from different locations.

For bug tracking purposes, our team uses Mantis Bug Tracking System. Mantis is a free web-based bug tracking system, which is written in the PHP scripting language and works with a database and a web server. It is released under the terms of the GNU General Public License. Team members may connect, using any web browser, to server on which Mantis is deployed, and report bugs or view previously reported ones in details.

Furthermore, NetBeans IDE, the development environment that our team uses; not only provides CVS support but also enables instant communication of team members via an add-in called Developer Collaboration.

3 The Configuration Management Process

3.1 Configuration Identification

The current state of ServerTheon will be identified by examining the following CIs:

- Sources
- Database Objects
- Documents

As identification process requires; specifications, naming conventions and acquisition details for each of above CIs are described below.

Sources

This CI includes class implementations for all of the main system modules and web interface modules described in Detailed Design Report. External auxiliary files that the compiled system will use, such as configuration files or 'cascading style sheets' (css files); are also a part of this CI. Classes are named in a manner that, name reflects the task of the class and only the first letter of each word is capitalized. This naming convention also holds for non-class files included in this CI. CVS will keep track of the files included in this CI; hence securing, storing, retrieving and reproducing are controlled by CVS.

Database Objects

Database table structure designed constitutes the content of this CI. Since this structure is kept by MySQL database server internally; instead of controlling the internal representation, a file containing SQL queries for constructing the table structure is used. This file is named as “CreateDatabaseTables.sql” and is again kept in CVS, which means that acquisition process will be controlled by CVS.

Documents

This CI involves all documents, prepared by TheonTech, that are subject to change such as living schedules, user manuals, installation manuals, release notes or software specifications. Documents will be stored, retrieved and updated using ‘Docs’ section in Mantis. For living schedules naming convention is as “TheonTech-LivingSchedule-vX-[DD-MM-YYYY].pdf”, where X is the version number and DD-MM-YY is the date the schedule updated. For other documents name will contain only the version number, i.e. “DocumentName-vX.FileExtension”.

3.2 Configuration Management and Control

3.2.1 Requesting changes

Requests involving major changes are made using Mantis interface, using a custom field in order not to be confused with bug issues, by using the following format:

- Urgency indicator [URGENT | NORMAL | CAN WAIT]
- Date and time of request [DD-MM-YYYY HH:MM]
- Version of the source code
- List of CIs and CI parts involved
- Name of requesting member

- The reason for change
- The description of change

On the other hand, minor changes in class implementations, such as little bug fixes or source code updates that do not affect other classes, can be made directly without request by the member or members responsible for that class. However, log of any such change will also be stored.

3.2.2 Evaluating changes

Change requests are evaluated weekly, but in case of emergencies some requests may be evaluated immediately, after the requesting member prepares the change request, in an online meeting arranged. Requests are ordered first with respect to urgency indicator and then with respect to date and time; most urgent request is evaluated first. In the case that reason for or description of change is thought to be unclearly or inadequately specified, CMC try to extend these specifications before evaluating the request. Evaluation involves discussions about reasons for change and impacts of the change to CIs. All change requests are tried to be evaluated each week; but if the time does not permit, at least URGENT ones are evaluated.

3.2.3 Approving or disapproving changes

According to evaluation results, a change request is either approved or disapproved. If approved; members who are responsible for implementation of the related classes are assigned to complete the requested change. Otherwise, no assignment is made and disapproval is logged. On the other hand, if change request involves CIs that are not related to class implementations, i.e. database objects CI or documents CI; one or more members are assigned for the change to be made. Furthermore, a time interval for the implementation of the change is decided and assigned.

3.2.4 Implementing changes

After implementation of a change is assigned, members who are responsible try to integrate changes to the CIs; in case of failure or requirement for additional change, they may prepare a new change request or just request participation of other members. When the implementation of the requested change is completed, related CIs are updated. Perhaps CIs other than the changed ones may be updated, i.e. a user manual update. Also, the implementation details are logged in the following format:

- List of associated change requests
- List of affected CIs [Name & Version]
- Date and time of assignment [DD-MM-YYYY - HH:MM]
- Date and time of completion [DD-MM-YYYY - HH:MM]
- List of involved member names
- Version for updated source
- List of changes applied [File : List-of-changes]

This format also applies to minor changes described in ‘Requesting changes’ part, excluding “list of associated change requests” and “date and time of assignment” entries.

3.3 Configuration Status Accounting

3.3.1 Change request reports

Change request reports will be generated weekly, for being used in evaluation process of change requests, using the reporting function in Mantis. These reports will contain the status information and the details of requested changes in the previous week and also implementation status of the change requests that were assigned to resources in the previous weekly meeting. After evaluation of the report is finished, the report will be stored in Docs section of Mantis with the name “ChangeRequestReport-wX-[DD-MM-YYYY]”, where X denotes the weekly meeting number.

3.3.2 Bug reports

Bug reports will be generated bi-weekly using the reporting function in Mantis. These reports will contain the status information and the details of bugs encountered that have been not yet resolved. The report will be stored in Docs section of Mantis with the name “BugReport-X-[DD-MM-YYYY]”.

3.3.3 Build reports

Build reports will be prepared bi-weekly. These reports will contain information about major changes in the functionality of ServerTheon and information about fixed bugs since the previous development snapshot. Generation of this report involves examining the differences between previous two bug reports and also discussing previous change request reports. The report will be stored in Docs section of Mantis, and also will be included in current development snapshot.

3.3.4 Release reports

Two release reports will be prepared, one for first and one for final release of ServerTheon. The reports will contain similar information to build report, but with a more detailed information about current functionality of ServerTheon. These reports will also be stored in Mantis.

3.4 Configuration Audits

Functional configuration audits will be conducted, after each development snapshot is built, before demonstration of the snapshot by CMC. These FCAs will mainly involve simple unit tests applied to ServerTheon. In addition, a much more sophisticated FCA will be conducted after the first release of ServerTheon. This FCA will actually involve application of unit and stress tests to the system, which will be clearly described in Test Specification Report.

Process audits will be the final parts of regular meetings in which some parts of CMP actually implemented, i.e. evaluating changes and approving or disapproving changes. Hence, CMC will be able to ensure, on the fly, that the defined process is consistently followed.

4 Project Schedules – CM Milestones

Scheduling project work is an essential element of project management. A project schedule makes clear to all participants when work is expected to be completed and also shows the time-related dependencies between different project tasks.

We have prepared TheonTech Living Schedule to achieve scheduling in ServerTheon. While preparing our living schedule, we have considered the implementation, testing, debugging and documentation requirements.

4.1 Living Schedule

The living schedule is persistent and contains both the current and future plans/schedules, allowing for both mission planning and current plan execution. TheonTech Living Schedule can be found in Appendix-A.

4.2 CM Milestones

Integrations of the classes and modules are considered as the milestones that give the opportunity to implement Configuration Identification. Testing, debugging and fixing are the main milestones that help us perform the Configuration Control. According to the feedback that is obtained at the end of the control period, we can decide whether we make version changes, or not. Finally, documentation and coding based on the pre-defined conventions give us the opportunity to implement CSA and FCA.

5 Project Resources

While developing ServerTheon according to CMP, our main tools will be CVS and Mantis. CVS will provide an environment for the team to work on the project source simultaneously. CVS will ensure that version updates will not make the previous versions unavailable. Mantis will help bug tracking is achieved in a robust manner.

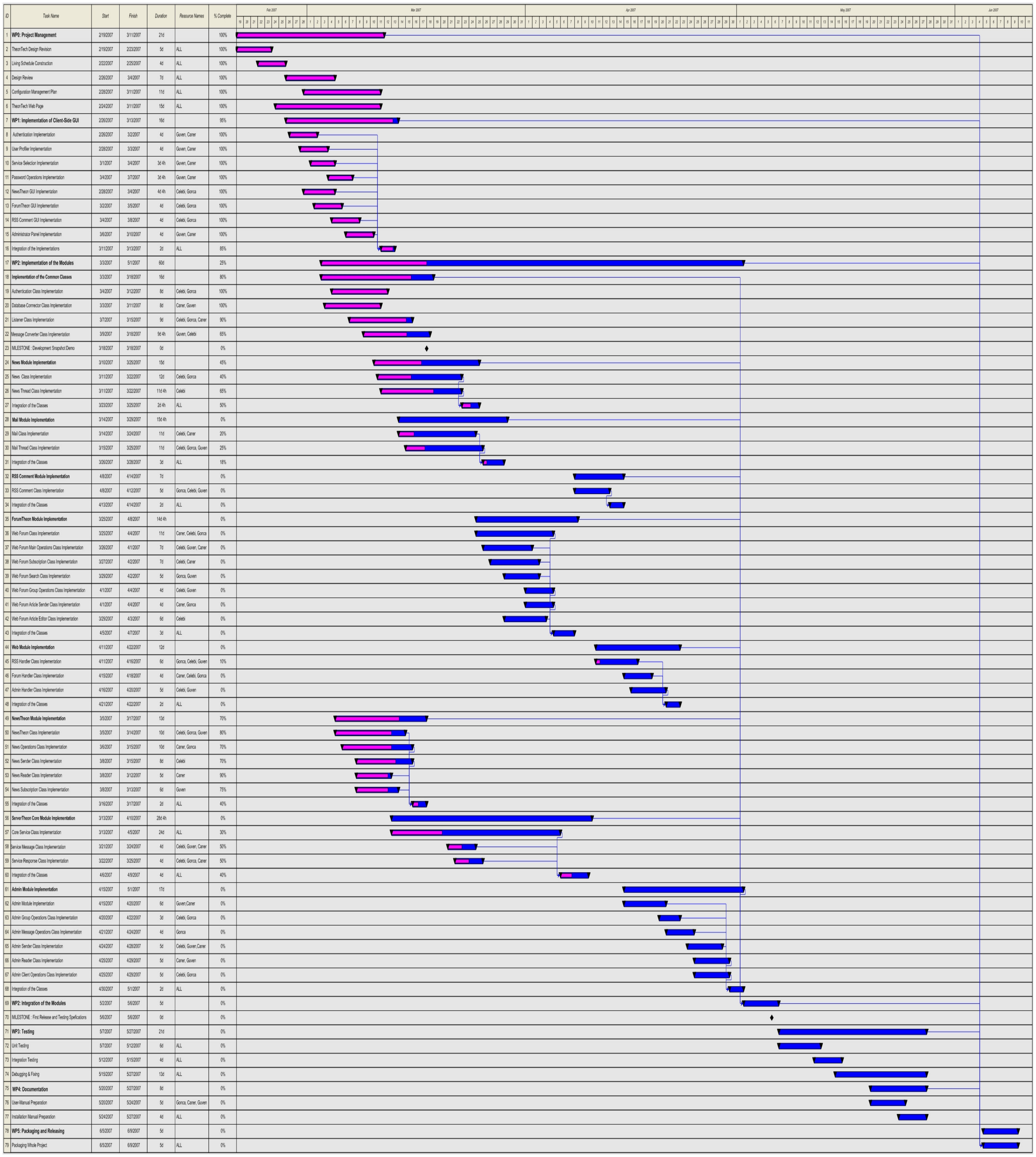
Documents that are part of CIs constitute another important resource for our project. We will follow the conventions described in CMP while developing ServerTheon which will be of great importance for completing the project.

Since development of the project is the common responsibility of all the project members, cooperation is vital. Developer Collaboration add-in in NetBeans will aid us in achieving this, which makes it another important resource.

6 Plan Optimization

We will need the optimization of ServerTheon CM Plan through the development and testing periods. We will have weekly meetings and demos during the spring term. At the end of each weekly meeting, we will come together and review ServerTheon. We will determine the necessary modifications and take necessary actions to keep ServerTheon CM Plan up-to-date. All of the TheonTech members will be responsible for plan optimization.

7 APPENDIX-A



ID	Task Name	Start	Finish	Duration	Resource Names	% Complete
1	**WP0: Project Management**	2/19/2007	3/11/2007	21d		100%
2	TheonTech Design Revision	2/19/2007	2/23/2007	5d	ALL	100%
3	Living Schedule Construction	2/22/2007	2/25/2007	4d	ALL	100%
4	Design Review	2/26/2007	3/4/2007	7d	ALL	100%
5	Configuration Management Plan	2/28/2007	3/11/2007	11d	ALL	100%
6	TheonTech Web Page	2/24/2007	3/11/2007	15d	ALL	100%
7	**WP1: Implementation of Client-Side GUI**	2/26/2007	3/13/2007	16d		95%
8	Authentication Implementation	2/26/2007	3/2/2007	4d	Guven, Caner	100%
9	User Profiler Implementation	2/28/2007	3/3/2007	4d	Guven, Caner	100%
10	Service Selection Implementation	3/1/2007	3/4/2007	3d 4h	Guven, Caner	100%
11	Password Operations Implementation	3/4/2007	3/7/2007	3d 4h	Guven, Caner	100%
12	NewsTheon GUI Implementation	2/28/2007	3/4/2007	4d 4h	Celebi, Gonca	100%
13	ForumTheon GUI Implementation	3/2/2007	3/5/2007	4d	Celebi, Gonca	100%
14	RSS Comment GUI Implementation	3/4/2007	3/8/2007	4d	Celebi, Gonca	100%
15	Administrator Panel Implementation	3/6/2007	3/10/2007	4d	Guven, Caner	100%
16	Integration of the Implementations	3/11/2007	3/13/2007	2d	ALL	85%
17	**WP2: Implementation of the Modules**	3/3/2007	5/1/2007	60d		25%
18	**Implementation of the Common Classes**	3/3/2007	3/18/2007	16d		80%
19	Authentication Class Implementation	3/4/2007	3/12/2007	8d	Celebi, Gonca	100%
20	Database Connector Class Implementation	3/3/2007	3/11/2007	8d	Caner, Guven	100%
21	Listener Class Implementation	3/7/2007	3/15/2007	9d	Celebi, Gonca, Caner	90%
22	Message Converter Class Implementation	3/9/2007	3/18/2007	9d 4h	Guven, Celebi	65%
23	MILESTONE : Development Snapshot Demo	3/18/2007	3/18/2007	0d		0%
24	**News Module Implementation**	3/10/2007	3/25/2007	15d		45%
25	News Class Implementation	3/11/2007	3/22/2007	12d	Celebi, Gonca	40%
26	News Thread Class Implementation	3/11/2007	3/22/2007	11d 4h	Celebi	65%
27	Integration of the Classes	3/23/2007	3/25/2007	2d 4h	ALL	50%
28	**Mail Module Implementation**	3/14/2007	3/29/2007	15d 4h		0%
29	Mail Class Implementation	3/14/2007	3/24/2007	11d	Celebi, Caner	20%
30	Mail Thread Class Implementation	3/15/2007	3/25/2007	11d	Celebi, Gonca, Guven	25%
31	Integration of the Classes	3/26/2007	3/28/2007	3d	ALL	18%
32	**RSS Comment Module Implementation**	4/8/2007	4/14/2007	7d		0%
33	RSS Comment Class Implementation	4/8/2007	4/12/2007	5d	Gonca, Celebi, Guven	0%
34	Integration of the Classes	4/13/2007	4/14/2007	2d	ALL	0%
35	**ForumTheon Module Implementation**	3/25/2007	4/8/2007	14d 4h		0%
36	Web Forum Class Implementation	3/25/2007	4/4/2007	11d	Caner, Celebi, Gonca	0%
37	Web Forum Main Operations Class Implementation	3/26/2007	4/1/2007	7d	Celebi, Guven, Caner	0%
38	Web Forum Subscription Class Implementation	3/27/2007	4/2/2007	7d	Celebi, Caner	0%
39	Web Forum Search Class Implementation	3/29/2007	4/2/2007	5d	Gonca, Guven	0%
40	Web Forum Group Operations Class Implementation	4/1/2007	4/4/2007	4d	Celebi, Guven	0%
41	Web Forum Article Sender Class Implementation	4/1/2007	4/4/2007	4d	Caner, Gonca	0%
42	Web Forum Article Editor Class Implementation	3/29/2007	4/3/2007	6d	Celebi	0%
43	Integration of the Classes	4/5/2007	4/7/2007	3d	ALL	0%
44	**Web Module Implementation**	4/11/2007	4/22/2007	12d		0%
45	RSS Handler Class Implementation	4/11/2007	4/16/2007	6d	Gonca, Celebi, Guven	10%
46	Forum Handler Class Implementation	4/15/2007	4/18/2007	4d	Caner, Celebi, Gonca	0%
47	Admin Handler Class Implementation	4/16/2007	4/20/2007	5d	Celebi, Guven	0%
48	Integration of the Classes	4/21/2007	4/22/2007	2d	ALL	0%
49	**NewsTheon Module Implementation**	3/5/2007	3/17/2007	13d		70%
50	NewsTheon Class Implementation	3/5/2007	3/14/2007	10d	Celebi, Gonca, Guven	80%
51	News Operations Class Implementation	3/6/2007	3/15/2007	10d	Caner, Gonca	70%
52	News Sender Class Implementation	3/8/2007	3/15/2007	8d	Celebi	70%
53	News Reader Class Implementation	3/8/2007	3/12/2007	5d	Caner	90%
54	News Subscription Class Implementation	3/8/2007	3/13/2007	6d	Guven	75%
55	Integration of the Classes	3/16/2007	3/17/2007	2d	ALL	40%
56	**ServerTheon Core Module Implementation**	3/13/2007	4/10/2007	28d 4h		0%
57	Core Service Class Implementation	3/13/2007	4/5/2007	24d	ALL	30%
58	Service Message Class Implementation	3/21/2007	3/24/2007	4d	Celebi, Guven, Caner	50%
59	Service Response Class Implementation	3/22/2007	3/25/2007	4d	Celebi, Gonca, Caner	50%
60	Integration of the Classes	4/6/2007	4/9/2007	4d	ALL	40%
61	**Admin Module Implementation**	4/15/2007	5/1/2007	17d		0%
62	Admin Module Implementation	4/15/2007	4/20/2007	6d	Guven,Caner	0%
63	Admin Group Operations Class Implementation	4/20/2007	4/22/2007	3d	Celebi, Gonca	0%
64	Admin Message Operations Class Implementation	4/21/2007	4/24/2007	4d	Gonca	0%
65	Admin Sender Class Implementation	4/24/2007	4/28/2007	5d	Celebi, Guven,Caner	0%
66	Admin Reader Class Implementation	4/25/2007	4/29/2007	5d	Caner, Guven	0%
67	Admin Client Operations Class Implementation	4/25/2007	4/29/2007	5d	Celebi, Gonca	0%
68	Integration of the Classes	4/30/2007	5/1/2007	2d	ALL	0%
69	**WP2: Integration of the Modules**	5/2/2007	5/6/2007	5d		0%
70	MILESTONE : First Release and Testing Spefications	5/6/2007	5/6/2007	0d		0%
71	**WP3: Testing**	5/7/2007	5/27/2007	21d		0%
72	Unit Testing	5/7/2007	5/12/2007	6d	ALL	0%
73	Integration Testing	5/12/2007	5/15/2007	4d	ALL	0%
74	Debugging & Fixing	5/15/2007	5/27/2007	13d	ALL	0%
75	**WP4: Documentation**	5/20/2007	5/27/2007	8d		0%
76	User-Manual Preparation	5/20/2007	5/24/2007	5d	Gonca, Caner, Guven	0%
77	Installation Manual Preparation	5/24/2007	5/27/2007	4d	ALL	0%
78	**WP5: Packaging and Releasing**	6/5/2007	6/9/2007	5d		0%
79	Packaging Whole Project	6/5/2007	6/9/2007	5d	ALL	0%