ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Glossary

Version <2.0>

Revision History

Date	Version	Description	Author
20/01/2001	01	First Draft	Sabrina Laflamme
06/04/2001	02	Adding new acronyms	Robert Latour Sabrina Laflamme

Table of Content

Glossary

1. Introduction

1.1 Purpose

The glossary contains the working definitions for all classes in TMT System. This glossary will be expanded throughout the life of the project.

1.2 Scope

This glossary addresses all terms which have specific meanings for this project. Actors are not listed here as they are described more fully in the use case definitions.

1.3 References

None.

2. Definitions, acronyms, and abbreviations

Allaire JRUN

Web Server used in the TMT project. Allow JSP and Servlet-based development.

Browser

A Browser is a software which allows the user to visualize and interact with all the information presented and flowing through the World Wide Web.

DCM

Developer Client Module

Engine

In a software or in a Computer, Engine is the term used for smaller programs executing specific functionalities and useful tasks for other programs or software

HTML

HTML (Hypertext Markup Language) is a text-based programming language using many symbols and codes interpreted statically by a web Browser.

Java

Java is a programming language mainly used in an Internet-Distributed environment. Java is based on C++ language but is much simpler to use and suggests an Object-Oriented model.

JDBC

Java Database Connectivity (JDBC) is an Application Program Interface (API) which links softwares written in Java to Databases JDBC allows database queries encoding and manages the returned results.

JSP (Java Server Pages)

A technology used to control dynamically the content of web pages by using small programs / applications called Servlet. The information is managed, controlled and changed on the web server first and then displayed in is final shape to the end-user.

MCM

Manager Client Module

SM

Server Module

Servlet

A Technology that generates dynamic web pages but differs from JSP: Servlets are precompiled JAVA code blocks.

Timestamp

A time unit (quantifiable) spent during a development activity related to the software / project. The time unit is qualified in terms of: week, length, project, task, activity and secondary activity.

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Software Requirements Specifications

Version <1.0>

Revision History

Date	Version	Description	Author
<01/01/2001>	<1.0>	First version	Martin Robillard

Table of Contents

Software Requirements Specifications

1. Introduction

1.1 Purpose

The purpose of this document is to describe the requirement specifications for a time monitoring tool for software development teams.

The intended audience of this document includes the prospective developers of the tool and the technical assessment personnel of the client organization.

1.2 Scope

The software system to be produced is a Time Monitoring Tool, which will be referred to as "TMT" thorough this document.

The Time Monitoring Tool will allow developers working within a defined software development process to record the time spent on the various software development activities, such as designing, coding, testing, or debugging. The TMT will also allow a manager to derive analyses and produce reports based on the data entered in the system.

The TMT could be used in any software development application to record the resources spent on the various software development activities. The objective of the TMT is to record the various activities being done. The goals are manifolds depending on the users of the data. The developers use TMT to record the activities that are being performed. The managers use data recorded from TMT to validate its planning, budgets and schedules. The software process manager use data from TMT to better understand the various software processes prescribed practices and to guide software process improvement concerns.

1.3 Definitions, acronyms, and abbreviations

Activity: nature of what is being done. The software process manager defines activities. Examples of activities are: analyzing, coding, designing, testing, etc.

Artifact: physical entity that results from an activity. Required artifacts are defined by the software engineering process. Examples of artifacts are SRS, architecture diagram, UML diagrams, source code, test scripts, user manuals.

DCM :	Developer Client Module
JSP :	Java Server Pages ,
MCM :	Manager Client Module
MCW:	Manager Client Window
SM:	Server Module
SRS:	Software Requirement Specification

Task: unit of work defined by the manager and for which the developer is accountable. task is a component of the schedule with a start and end date.

Timestamp: A qualified unit of time spent on a software development-related activity. The unit is qualified in terms of the following pieces of information: week, duration, project, task, activity, and sub-activity.

TMT: Time Monitoring Tool, which is the name of the tool to be developed

TMW: Time Monitoring Window

1.4 References

IEEE Std. 830-1993: IEEE Recommended Practice for Software Requirements Specifications

JSP: http://www.esperanto.org.nz/jsp/jspfaq.jsp#q1

1.5 Overview

The rest of this document contains an overall description of the Time Monitoring Tool software system (section 2), and the specific requirements for the system (section 3).

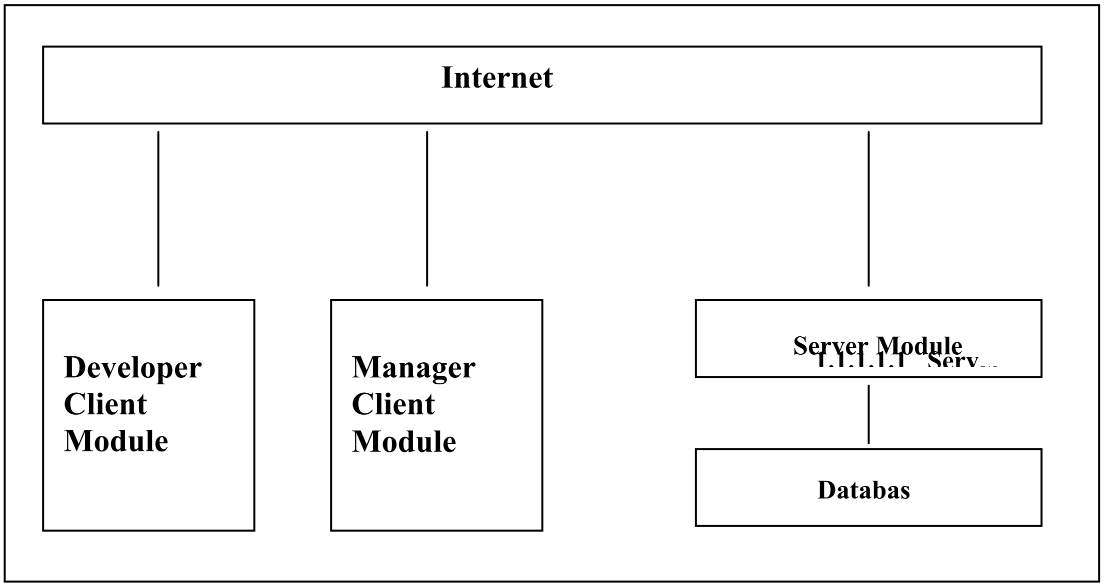
2. Overall description

2.1 Product perspective

In some software development organizations, developers are required to record the time spent on the various activities that are related to the development of software. This information is commonly used for various administration tasks, such as budget planning. Without tool support, precisely recording the time spent on each activity can be a tedious task. A time monitoring tool can ease this task.

2.1.1 System Interfaces

The TMT system to be developed is a stand-alone tool that is integrated within the organization's Intranet. It consists in four major components: a Developer Client Module, a Server Module, a Database, and a Manager Client Module (see diagram 1).



The DCMallows developers to log onto the TMT system and to record timestamps corresponding to their activities in a convenient way. The server module is a daemon that accepts connections from Developer and Manager Client Modules and serves as an interface between these modules and a database. The Manager Client Module allows a manager to retrieve timestamp information from the database to produce analyses and reports. The Database component can be any type of database. It does not have to be developed as a part of the TMT system, as long as the Server Module can interface with an available database system.

All components must execute on WindowsNT.

2.1.2 User interfaces

The Developer and Manager Client Modules must provide a user interface that is available through the Netscape WWW browser. The Server Module does not have a user interface other than a command to launch it. The Database module does not have a user interface.

2.1.3 Hardware interfaces

All components must be able to execute on a personal computer.

2.1.4 Software Interfaces

The Developer and Manager Client Modules must be Java Applets running within Netscape. The Server Module mu st integrate with a DBMS through the Java Database Connectivity (JDBC) interface. The Server must run within a Web Server available for WindowsNT, using dynamic content technology (e.g., Java Server Pages (JSP), CGI/Perl, and Cold Fusion).

2.1.5 Communication interfaces

The Developer and Manager Client Modules must communicate with the server over a TCP/IP connection. The Server and the Database components should be located on the same host.

2.1.6 Memory Constraints

The Developer and Manager Clients must be able to operate within 64MB (including memory for Netscape), and the server and database within 128MB.

2.1.7 Operations

The operation of the Developer and Manager client Modules must be easy and intuitive for professional software developers and mangers. No specific formation must be required to use the tool. The Server module must be installed and maintained with no interaction with existing software and should not required any specific technical skill from a network administrator.

Back up operations must be defined.

Recovery operations must be specified in case of network failure, user machine failure and database failure.

2.1.8 Site adaptation requirements

No specific site adaptation should be required.

2.2 Product functions

The two main functions of the Time Monitor software system are to allow developers to use a www browser to store timestamp records in a database, and to allow a manager to analyze these timestamp records.

A timestamp record consists of the time duration of a specific activity with its unique identification. The unique identification is made of three components: the project, the user and the date when the activity is taken place. The description of an activity is divided into three components: a task name, an activity, and an artifact.

For managerial purposes it is often useful to define the date in terms of the current week. The current week is defined as the week starting on the Monday immediately preceding the current day of the week, and ending on the Sunday immediately following the current day of the week, inclusively.

A task is a unit of work defined by the manager and for which the developer is accountable. A task is a component of the schedule with a start and end date. Examples of tasks are Implement module A, Design library XYZ. Developers usually work on assigned tasks. One developer may work on many tasks and a given task may involve many developers.

An activity is the nature of what is being done. Software developers are mostly involved in cognitive activities. The various cognitive activities composed the practices that are defined from the software engineering processes. The software process manager defines the activities and the developer must select the activities that are most representative of what is being done during the recorded time. Typical examples of activities are analyzing, coding, designing, testing, etc.

An artifact is a physical entity that results from an activity. The software engineering process defines required artifacts . Typical artifacts are SRS, architecture diagram, UML diagrams, source code, test scripts, user manuals, etc....Developers must select artifacts among predefined lists.

Names of Tasks, Activities and Artifacts are predefined for a project and are stored in the database of the TMT system.

The database stores two different types of items: user configurations and time stamp records. User configurations consist in a username, a password, and a list of projects, tasks, activities and artifacts currently available for this user.

Time stamp records stored in the database consist of a time stamp as described above, plus a field identifying the week, and a field identifying the username.

2.3 User characteristics

Users are software developers familiar with the web technology. Users are knowledgeable of the software engineering process and have good understanding of the tasks, activities and artifacts they are being involved with.

2.4 Constraints

The system should enforce user authentication security and guaranty timestamps reliability.

2.5 Assumptions and dependencies

No specific assumption or dependencies.

3. Specific functional requirements

3.1 Developer Client Module (DCM)

3.1.1 The user shall be able to load the DCM within Netscape

3.1.2 The DCM shall support the logging of users.

3.1.2.1 The initial window of the DCM shall contain a field for a user name, a field for a password, and a button labeled login. The password field shall be a "secret" field, which does not display what the user types.

3.1.2.2 When a user presses the login button, the DCM shall send a request to the SM to login the user.

3.1.3 If the logging of a user is successful (see 3.1.2.2), the DCM shall display the TMW.

3.1.3.1 The TMW shall always display the identifier for the current week.

3.1.3.2 The TMW shall always display the username of the user currently logged in.

3.1.3.3 The TMW shall display all the validated and non-validated time stamp records previously entered for the current week by the user currently logged in.

3.1.3.3.1 The validated and non-validated time stamp records shall be displayed in a tabular fashion, with every record being on a separate row.

3.1.3.3.2 The fields should be identified with labels. Intuitive and non-confusing abreviations can be used if necessary.

3.1.3.3.3 The fields should be non modifiable to prevent modification of the time stamp records.

3.1.3.4 The TMW shall display all the refused time stamp records previously entered by the user currently logged in.

3.1.3.4.1 The refused time stamp records shall be displayed in a tabular fashion, with every record being on a separate row.

3.1.3.4.2 The fields should be identified with labels. Intuitive and non-confusing abreviations can be used if necessary.

3.1.3.4.3 The fields should me modifiable to allow correction of the time stamp records.

3.1.4 The existing TMW timestamps shall be updateable

3.1.4.1 It shall only be possible to modify the Project, Task, Activity, and Artifacts fields with values that are allowed for the user currently logged in.

3.1.4.2 It shall only be possible to modify the time fields for every day of the week using a positive numerical value of maximum 24.

3.1.4.3 Modifications to the records shall not be synchronized automatically with the server, that is, it should be possible to modify several values and then to click a bottom or perform another mechanism to update the server.

3.1.5 The TMW shall support the entry of new timestamps

3.1.5.1 It shall be possible to add new timestamps according to predefined fields

3.1.5.2 All new timestamps should come with the predefined selections for a given user.

3.1.5.3 It shall be possible to create up to 100 new timestamps for a given user and a given current week.

3.1.5.4 All time entry shall be in hours and minutes H:mn

3.1.6 The TMW shall support entry of timestamps

3.1.6.1 All fields of a timestamp shall have predefined values for the logged in user.

3.1.6.2 Project, Task, Activity, Artifact and at least one time field must be filled out before the record is sent to the database.

3.1.6.3 Time entry could be in duration or taxi mode.

3.1.6.4 G clock for the taxi mode.

3.1.7 The TMW shall provide cumulative totals

3.1.7.1 Each column day of the current week shall have the total number of hours.

3.1.7.2 Each project and task line of the current week shall have the total number of ours recorded.

3.1.7.3 The total number of hours recorded in the current week shall be displayed

3.1.7.4 All time records shall be displayed in hours and minutes: H:mn.

3.2 Manager Client Module (MCM)

3.2.1 The manager shall be able to load the Manager Client Module within Netscape

3.2.2 The Manager Client Module shall support the logging of managers.

3.2.2.1 The initial window of the MCM shall contain a field for a user name, a field for a password, and a button labeled login. The password field shall be a "secret" field, which does not display what the user types.

3.2.2.2 When a user presses the login button, the MCM shall send a request to the SM to login the user.

3.2.3 If the logging of a user is successful (see 3.2.2.2), the MCM shall display the MCW.

3.2.3.1 The MCW shall always display the username of the manager currently logged in.

3.2.3.2 The MCW shall display two groups of icons, which are for the user management and the project management.

3.2.4 MCW shall support user's management

3.2.4.1 The user management icons shall included: add a user, display/modify/disable users, and validate timestamps.

3.2.4.2 Clicking add user icons shall display a fill in form for adding a user to the project

3.2.4.2.1 Add user form shall enable the recording of the user id, the projects, the user's supervisor identification, and the selection of predefined fields for this user.

3.2.4.2.2 Add user form shall be validated for completeness before being sent to the Server Module.

3.2.4.3 Clicking Display all users icon should display the list of all users in alphabetical order with the identification of the projects that they are involved in and their supervisor.

3.2.4.3.1 This option could provide a list ordered alphabetically by last name of the user or by project or by supervisor.

3.2.4.3.2 By selecting a user, his profil can be changed or disabled.

3.2.4.4 Clicking validate timestamps should display the list of all timestamps records submit by users.

3.2.4.4.1 The timestamp records shall be displayed in a tabular fashion, every record being on a separate row.

3.2.4.4.2 The fields should be identified with labels. Intuitive and non-confusing abreviations can be used if necessary.

3.2.4.4.3 The manager can validate a timestamp record, refuse a record or leave timestamp record non-validated.

3.2.4.4.4 The validated timestamps records should be inserted in the database.

3.2.4.4.5 The refused timestamps records should be returned to the user to allow correction.

3.2.4.4.6 The non-validated timestamps record should be stayed in the table to allow manager to validate or refuse a timestamp record another time.

3.2.5 MCW shall support project's management

3.2.5.1 The project's management icon shall include the add project, add task, MS EXCEL and MSProject icons.

3.2.5.2 Clicking add project icon shall display a form to identify the new project to add.

3.2.5.3 Clicking the add task icon shall display a form to add a new task in a project selected.

3.2.5.4 Clicking the EXCEL icon shall display a form to identify the project; users, week's data and the EXCEL file where the corresponding data from the database should be transferred.

3.2.5.5 Clicking the MSProject icon shall display a form to identify the project; users, week's data and the MSProject file where the corresponding data from the database should be transferred.

3.3 Server Module (SM)

3.3.1 The SM shall be the only intermediate between the two client modules and the database.

3.3.2 The SM shall receive all the requests and format the pages.

3.3.3 The SM shall accept all connections from developers and manager client modules.

3.3.4 Upon log in request from the DCM the SM shall produce the TMW as specified in 3.1.2

3.3.5 Upon request for updating time stamp records from the DCM the SM shall update the database to reflect the new state of the DCM.

3.3.6 The SM shall validate and execute all requests coming from the DCM.

3.3.7 Upon log in request from the MCM the SM shall produce the MCW as specified in 3.2.2.

3.3.8 The SM shall validate and execute all requests coming from the MCM.

3.3.9 The SM shall display the TMT status

3.3.9.1 Any error of execution, communication, validation or else shall be identified and appropriate comment display.

3.3.9.2 The SM shall try to recovery from most common errors.

4. Classification of functional requirements

Functionality	Type
3.1.1 The user shall be able to load the DCMwithin Netscape	Essential
3.1.2 The DCMshall support the logging of users.	Essential
3.1.2.1 The initial window of the DCM shall contain a field for a user name, a field for a password, and a button labeled login. The password field shall be a "secret" field, which does not display what the user types.	Essential
3.1.2.2 When a user presses the login button, the DCM shall send a request to the SM to login the user.	Essential
3.1.3 If the logging of a user is successful (see 3.1.2.2), the DCM shall display the TMW.	Essential
3.1.3.1 The TMW shall always display the identifier for the current week.	Desirable
3.1.3.2 The TMW shall always display the username of the user currently logged in.	Desirable
3.1.3.3 The TMW shall display all the time stamp records previously entered for the current week by the user currently logged in.	Essential
3.1.3.3.1 The time stamp records shall be displayed in a tabular fashion, with every record being on a separate row.	Desirable
3.1.3.3.2 The fields should be identified with labels. Intuitive and non-confusing abbreviations can be used if necessary.	Desirable
3.1.4 The existing TMW timestamps shall be updateable.	Essential
3.1.4.1 It shall only be possible to modify the Project, Task, Activity, and Artifacts fields with values that are allowed for the user currently logged in.	Essential
3.1.4.2 It shall only be possible to modify the time fields for every day of the week using a positive numerical value of maximum 24.	Essential
3.1.4.3 Modifications to the records shall not be synchronized automatically with the server, that is, it should be possible to modify several values and then to click a bottom or perform another mechanism to update the server.	Essential
3.1.5 The TMW shall support the entry of new timestamps.	Essential
3.1.5.1 It shall be possible to add new timestamps according to predefined fields.	Desirable

3.1.5.2 It shall be possible to create up to 100 new timestamps for a given user and a given current week.	Essential
3.1.5.3 All new timestamps should come with the predefined selections for a given user.	Desirable
3.1.5.4 All time entry shall be in hours and minutes H:mn.	Essential
3.1.5.5 All empty time fields shall be at 0	Desirable
3.1.6 The TMW shall support entry of timestamps	Essential
3.1.6.1 All fields of a timestamp shall have predefined values for the logged in user.	Desirable
3.1.6.2 Project, Task, Activity, Artifact and at least one time field must be filled out before the record is sent to the database.	Essential
3.1.6.3 Time entry could be in duration or taxi mode.	Desirable
3.1.6.4 TMW shall provide a running clock for the taxi mode.	Desirable
3.1.7 The TMW shall provide cumulative totals.	Essential
3.1.7.1 Each column day of the current week shall have the total number of hours recorded.	Essential
3.1.7.2 Each project and task line of the current week shall have the total number of hours recorded.	Essential
3.1.7.3 The total number of hours recorded in the current week shall be displayed.	Essential
3.1.7.4 All time records shall be displayed in hours and minutes: H:mn.	Essential
3.2.1 The manager shall be able to load the Manager Client Module within Netscape.	Essential
3.2.2 The Manager Client Module shall support the logging of managers.	Essential
3.2.2.1 The initial window of the MCM shall contain a field for a user name, a field for a password, and a button labeled login. The password field shall be a "secret" field, which does not display what the user types.	Essential
3.2.2.2 When a user presses the login button, the MCM shall send a request to the SM to login the user.	Essential
3.2.3 If the logging of a user is successful (see 2.2.2), the MCM shall display the MCW.	Essential
3.2.3.1 The MCW shall always display the username of the manager currently logged in.	Desirable
3.2.3.2 The MCW shall display two groups of icons, which are for the user management and the project management.	Desirable

3.2.4 MCW shall support user's management.	Essential
3.2.4.1 The user management icons shall included: add a user, delete a user, display all users, and modify user's file.	Essential
3.2.4.2 Clicking add user icons shall display a fill in form for adding a user to the project.	Essential
3.2.4.2.1 Add user form shall enable the recording of the user id, the projects, the user's supervisor identification, and the selection of predefined fields for this user.	Essential
3.2.4.2.2 Add user form shall be validated for completeness before being sent to the SM.	Essential
3.2.4.3 Clicking delete user icons shall display the list of all users in alphabetical order with a check box for delete user and the date of the day that could be overwritten.	Essential
3.2.4.4 Clicking Display all users icon should display the list of all users in alphabetical order with the identification of the projects that they are involved in and their supervisor.	Essential
3.2.4.4.1 Display all users could provide a list ordered alphabetically by last name of the user or by project or by supervisor.	Desirable
3.2.4.5 Clicking Modify user's file icon shall display the list of users in alphabetical order with a check box for modify user's file.	Essential
3.2.4.5.1 Clicking the modify user file shall display the complete user file an enable overwriting of any field.	Essential
3.2.5 MCW shall support project's management	Optional
3.2.5.1 The project's management icon shall include the EXCELL and MSProject icons.	Optional
3.2.5.2 Clicking the EXCELL icons shall display a form to identify the project; users, week's data and the EXCELL file where the corresponding data from the database should be transferred.	Optional
3.2.5.3 Clicking the MSProject icons shall display a form to identify the project; users, week's data and the MSProject file where the corresponding data from the database should be transferred	Optional
3.2.5.4 All forms shall be sent to SM for validation.	Essential
3.2.5.4.1 Unvalidated fields shall be clearly indicated with appropriate comments.	Desirable
3.3.1 The SM shall be the only intermediate between the two client modules and the database.	Essential

3.3.2 The SM shall receive all the requests and format the pages.	Essential
3.3.3 The SM shall accept all connections from developers and manager client modules.	Essential
3.3.4 Upon log in request from the DCM the SM shall produce the TMW as specified in 3.1.2.	Essential
3.3.5 Upon request for updating time stamp records from the DCM the SM shall update the database to reflect the new state of the DCM.	Essential
3.3.6 The SM shall validate and execute all requests coming from the DCM.	Essential
3.3.7 Upon log in request from the MCM the SM shall produce the MCW as specified in 3.2.2	Essential
3.3.8 The SM shall validate and execute all requests coming from the MCM.	Essential
3.3.9 The SM shall display the TMT status.	Essential
3.3.9.1 Any error of execution, communication, validation or else shall be identified and appropriate comment display.	Desirable
3.3.9.2 The SM shall try to recovery from most common errors.	Desirable

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Supplementary Specifications

Version <2.0>

Revision History

Version	Date	Description	Author
1.0	16/01/2001	First Draft	Sabrina Laflamme
2.0	05/02/2001	Applied modifications as requested by the Lab Attendant	Sabrina Laflamme

Table of Content

Supplementary Specifications

1. Introduction

1.1 Purpose

Supplementary specifications capture the requirements which aren't easily defined within the UseCase Model. Requirements such as: legal standards, quality aspects, reliability, supportability, and execution criteria of the system.

1.2 Definitions, acronyms, and abbreviations

Browser

A Browser is a software which allows the user to visualize and interact with all the information presented and flowing through the World Wide Web.

DCM

Developer Client Module

Engine

In a software or in a Computer, Engine is the term used for smaller programs executing specific functionalities and useful tasks for other programs or software

JSP (Java Server Pages)

A technology used to control dynamically the content of web pages by using small programs / applications called Servlet. The information is managed, controlled and changed on the web server first and then displayed in is final shape to the end-user.

MCM

Manager Client Module

SM

Server Module

Timestamp

A time unit (quantifiable) spent during a development activity related to the software / project. The time unit is qualified in terms of: week, length, project, task, activity and secondary activity.

1.3 References

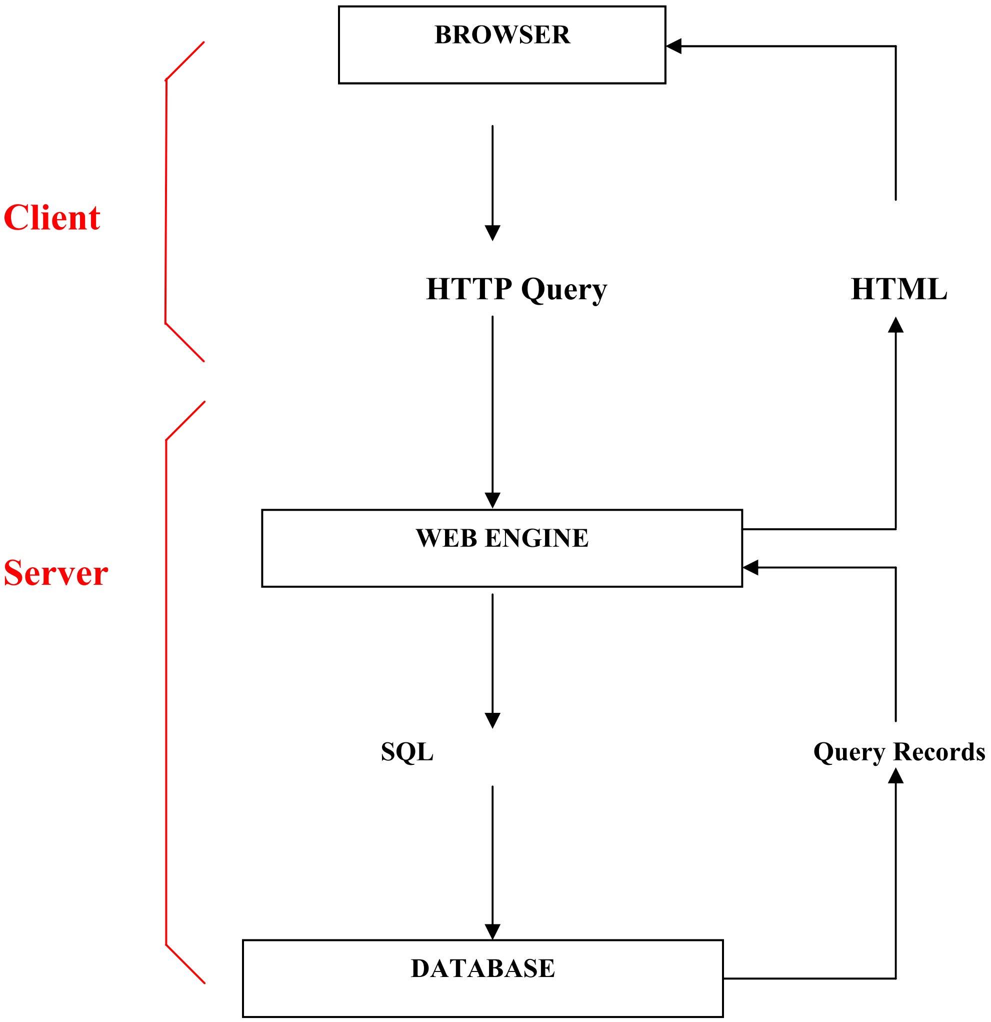
Course Web Page (INF4302) : http://www.cours.polymtl.ca/inf4302/

UPEDU : http://www.upedu.org/

2. Supplementary Specifications

2.1 Client / Server

The user interacts with the system through a web browser (Netscape or Internet explorer, Client). The browser executes HTML and JAVA queries to the Web Engine through JSP technology. The Engine communicates with the Database using SQL language. Finally the Database returns the data to the Engine which, in turn, gives back the result package to the browser. The browser then shows the results to the end-user.



2.2 Profitability

2.2.1 Graphical Interface

All interactions between human and computer are made through a graphical interface. Each system functionality must be accessible by mouse or keyboard. The user must be able to choose a specific branch from the system and then obtaining the results on the screen.

2.2.2 User's Knowledge

Targeted users are familiar with the use of web Browsers (Especially with Netscape Navigator or Internet Explorer) on a Windows-based platform. The tool is designed to be user-friendly and the user shall not need training.

2.3 Liability

The application shall be able to capture, execute and respond to all user-entries. All false entries shall be validated and returned to the user without stopping abnormally the application.

2.4 Performance

The application shall run on a Pentium processor. The application will execute itself on a Windows-based platform and response time for all tasks shall be inferior to 1.5 seconds.

2.5 Supportability

Client side can be installed on any platform but the Server side shall be installed on a Windows-Based platform.

2.6 Design Constraints

Programming language used: Java, JSP and HTML for all web content; SQL for all database management queries.

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Use-Case Specifications

Version <5.0>

Revision History

Date	Version	Description	Author
15-01-2001	1.0	First Draft	James Prevost
15-01-2001	2.0	Use-Case Diagrams added	James Prevost
14-02-2001	3.0	Use-Case Specifications finished	James Prevost
16-02-2001	4.0	Major revision	James Prevost
30-03-2001	5.0	Final Revision and addition of the « Timestamp Validation » functionality	James Prevost

Preface

The following case study has been modified from its original content. The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Requirements Section.

Regards,

Unified Process for Education Team

Table of Content

Figures

Use-Case Specifications

1. Use-Case Model

1.1 Introduction

The system involves Internet-Distributed Timestamps. Thus, it must be available from any remote terminal that might be used by the targeted users. User types are: *Developer* and Administrator. A user accesses the system by providing his username and password which were previously defined by the system administrator. The *Developer* uses the system though a Graphical User Interface (GUI) hosted on a web site from which he can fill in his latest timestamps. Developers are also required to modify mistaken timestamps identified by the *Administrator.* The *Administrator* also uses the system through a similar GUI but offered options are different: Creation of new users, modification of an existing user profile, timestamps sorting and listing, timestamps exportation and importation to/from external applications. All data is stored in a database accessed by Server Module upon execution of queries.

1.2 General Actors Descriptions

1.2.1 Developer

An employee. Enters timestamps in the database for validation by the *Administrator.* Might be required to modify or mistaken old entries. Has a personal username and password to access the system.

1.2.2 Administrator

The administrator mission is to manage all entered timestamps in the system. The administrator creates the usernames and passwords for other system users. Moreover, the administrator can list and sort the timestamps using different criteria, can import or export timestamps data from/to external applications such as Microsoft Excel and Microsoft Project.

1.2.3 DBMS

The Database Management System is an external actor. It stores data in the database and executes internal database queries.

1.2.4 MS EXCEL

Microsoft EXCEL is an external actor. It is one of the locations where data is exported to or imported from. Exportation involves the creation of a statistical spreadsheet or report. Importation involves adding new database entries from a spreadsheet.

1.2.5 MS PROJECT

Microsoft PROJET is an external actor. It is one of the locations where data is exported to or imported from. Exportation involves the creation of Gantt Diagrams. Importation involves adding new database entries from a Gantt Diagram.

1.3 Use-Case Model Hierarchy

1.3.1 Mngmt - Connections

- **Description**

This package is only a container package. It references to the sub packages mentioned in the "packages owned" list point.

- **Use Cases**
  - **Load MCM**
  - **Load DCM**
  - **Login**
  - **Logout**
  - **Show TMT Status**
  - **Identify errors**
  - **Correct errors**
  - **Show TMT Window**
  - **Close TMT Window**

- **Actors**
  - **Administrators**
  - **Developers**
  - **DBMS (Database Management System)**

- **Relationships**

This package is only a container package. It references to the sub packages mentioned in the following list point.

- **Packages Owned**
  - **Load Client Modules**
  - **Login/Logout Mngmt**

1.3.1.1 Packages Diagram

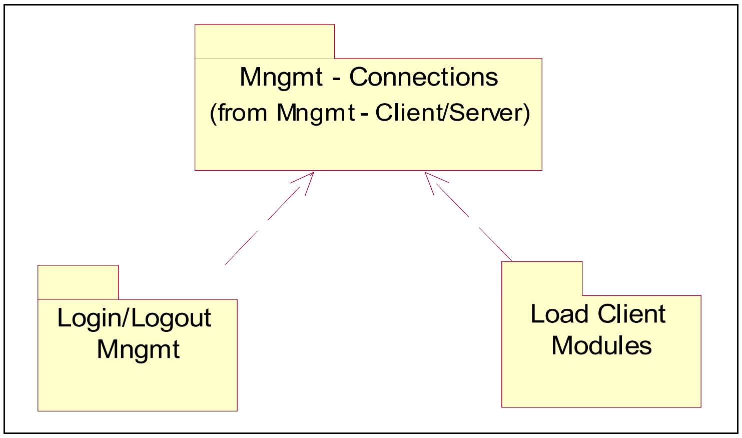


Figure 1 Mngmt – Connections / Use Packages

1.3.1.2 Load Client Modules

- **Description**
  Activates a Client Session

- **Use Cases**
  - **Load MCM**
  - **Load DCM**

- **Actors**
  - **Administrator**
  - **Developer**

1.3.1.3 Login/Logout Mngmt

- **Description**
  All users' connections management functionalities

- **Use Cases**
  - **Login**
  - **Logout**
  - **Show TMT Status**
  - **Identify errors**
  - **Correct errors**
  - **Show TMT Window**
  - **Close TMT Window**

- **Actors**
  - **Administrator**
  - **Developer**
  - **SGDB**

1.3.2 Mngmt - Developer

- **Description**

This package contains all management functionalities that a developer can exert.

- **Use Cases**
    - **Add Timestamp**
    - **Update Timestamp**
    - **Validate Query**
    - **Execute Query**
    - **Access Database**

- **Actors**
    - **Developer**
    - **DBMS (Database Management System)**

- **Relationships**
    - **None**

- **Packages Owned**
    - **None**

1.3.2.1 Packages Diagram

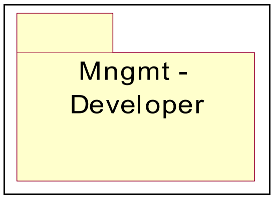


Figure 2 Mngmt - Developer

1.3.3 *Mngmt - Administrator*

- **Description**

This package contains all management functionalities that an administrator can exert.

- **Use Cases**
  - **Deactivate User**
  - **Add User**
  - **Validate Timestamps**
  - **Modify User**
  - **List Alphabetically**
  - **List by Project**
  - **List by Supervisor**
  - **Export to EXCEL**
  - **Export to MSPROJECT**
  - **Add Project**
  - **Add Task**
  - **Validate Query**
  - **Execute Query**
  - **Access Database**

- **Actors**
  - **Administrator**
  - **EXCEL**
  - **MSPROJECT**
  - **DBMS (Database Management System)**

- **Relationships**
  - **None**

- **Packages Owned**
  - **Developers Mngmt**
  - **External Data Mngmt**
  - **Project Mngmt**

1.3.3.1 Packages Diagram

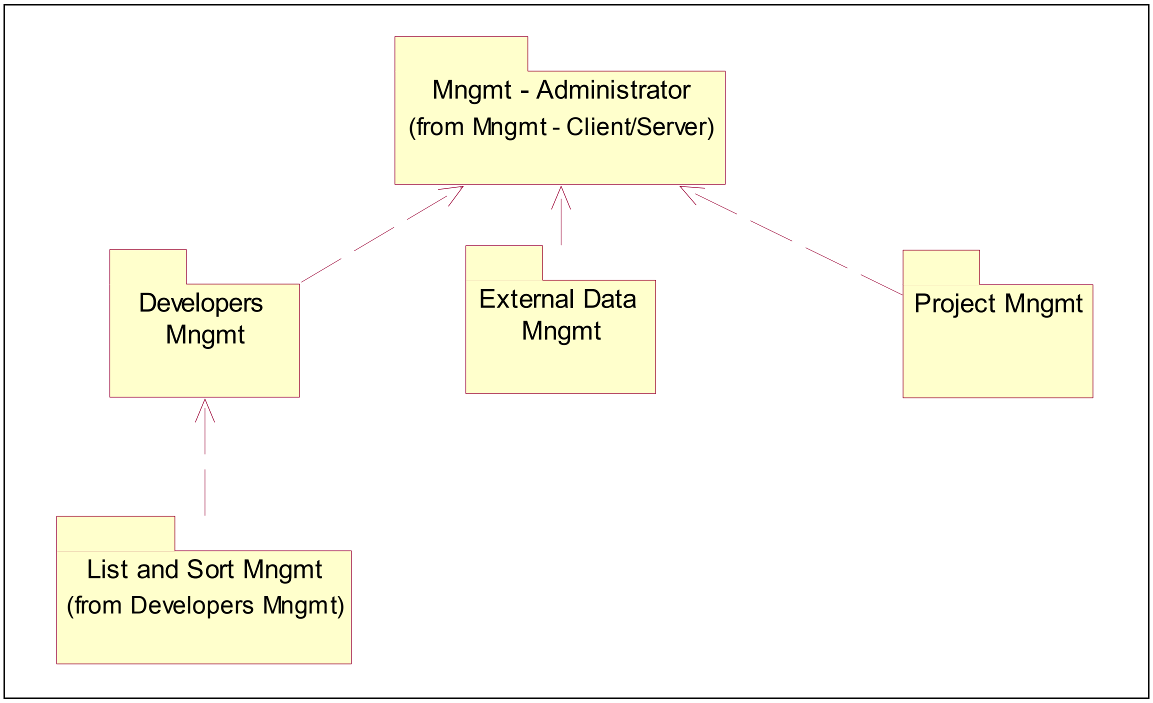


Figure 3 Mngmt - Administrator

1.3.3.2 Developers Mngmt

- **Description**

This package contains all developer management functionalities that an administrator can exert.

- **Use Cases**
  - **Deactivate User**
  - **Add User**
  - **Validate Timestamps**
  - **Modify User**
  - **List Alphabetically**
  - **List by Project**
  - **List by Supervisor**
  - **Validate Query**
  - **Execute Query**
  - **Access Database**

- **Actors**
  - **Administrator**
  - **DBMS (Database Management System)**

- **Relationships**
    - **None**

- **Packages Owned**
    - **List and Sort Mngmt**

List and Sort Mngmt

- **Description**

This package contains all developers' lists management functionalities that an administrator can exert.

- **Use Cases**
    - **List Alphabetically**
    - **List by Project**
    - **List by Supervisor**
    - **Validate Query**
    - **Execute Query**
    - **Access Database**

- **Actors**
    - **Administrator**
    - **DBMS (Database Management System)**

- **Relationships**
    - **None**

- **Packages Owned**
    - **None**

1.3.3.3 External Data Mngmt

- **Description**

This package contains all data management functionalities that an administrator can exert.

- **Use Cases**
    - **Export to EXCEL**
    - **Export to MSPROJECT**
    - **Validate Query**
    - **Execute Query**
    - **Access Database**

- **Actors**
    - **Administrator**
    - **EXCEL**
    - **MSPROJECT**
    - **DBMS (Database Management System)**

- **Relationships**
  - **None**

- **Packages Owned**
  - **None**

1.3.3.4 Project Mngmt

- **Description**

This package contains all project management functionalities that an administrator can exert.

- **Use Cases**
  - **Add Project**
  - **Add Task**
  - **Validate Query**
  - **Execute Query**
  - **Access Database**

- **Actors**
  - **Administrator**
  - **DBMS (Database Management System)**

- **Relationships**
  - **None**

- **Packages Owned**
  - **None**

1.4 Diagrams of the Use-Case Model

Diagrams of the use case model (use-case diagrams) are presented in each Section (Package) of use cases below.

Load Client Modules

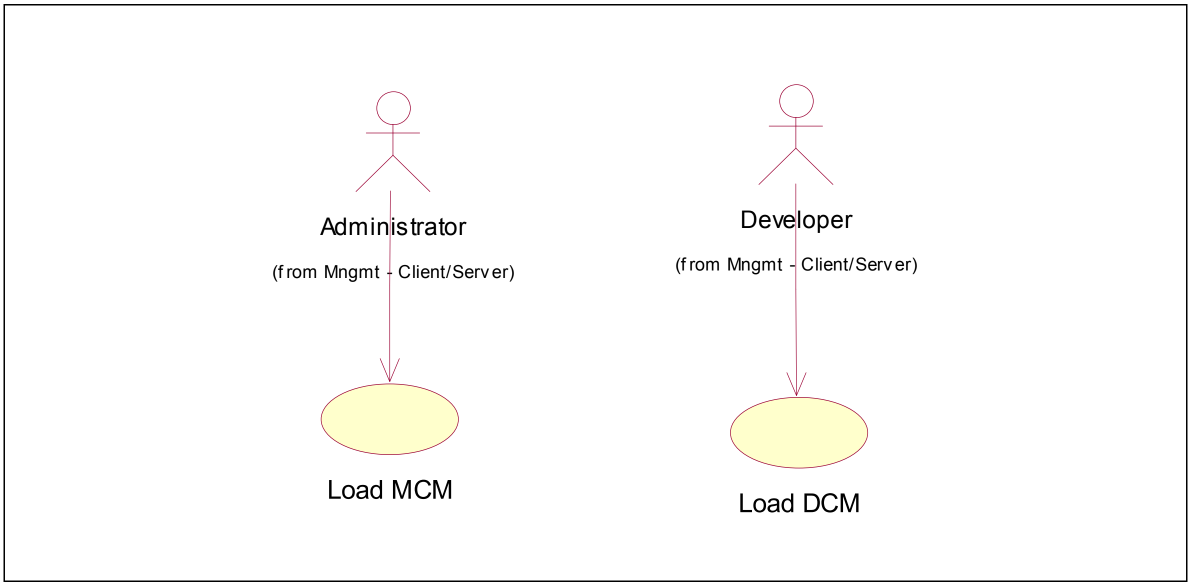


Figure 4 Load Client Modules

2. Load MCM

2.1 Brief Description

Loading the Manager Client Module so the *Administrator* can access the options the system allows him to perform. The loading of the module is made through a Browser.

2.2 Flow of Events

2.2.1 Basic Flow

After entering correct username and password, the Browser ensures that the MCM is loaded correctly and shows the corresponding GUI to the A*dministrator*

2.2.2 Alternative

None.

2.3 Preconditions

2.3.1 <Internet Connection>

The user must be connected to the internet.

2.3.2 <Java Technology Compatible Browser>

The user must use a Java-Supported Browser such as Internet Explorer or Netscape Navigator

2.4 Postconditions

None.

2.5 Extension Points

None.

2.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 4)

2.7 Other Diagrams

None.

3. Load DCM

3.1 Brief Description

Loading the Developer Client Module so the *Developer* can access the options the system allows him to perform. The loading of the module is made through a Browser.

3.2 Flow of Events

3.2.1 Basic Flow

After entering correct username and password, the Browser ensures that the DCM is loaded correctly and shows the corresponding GUI to the *Developer*

3.2.2 Alternative

None.

3.3 Preconditions

3.3.1 <Internet Connection>

The user must be connected to the internet.

3.3.2 <Java Technology Compatible Browser>

The user must use a Java-Supported Browser such as Internet Explorer or Netscape Navigator

3.4 Postconditions

None.

3.5 Extension Points

None.

3.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 4)

3.7 Other Diagrams

None.

Login/Logout Mngmt

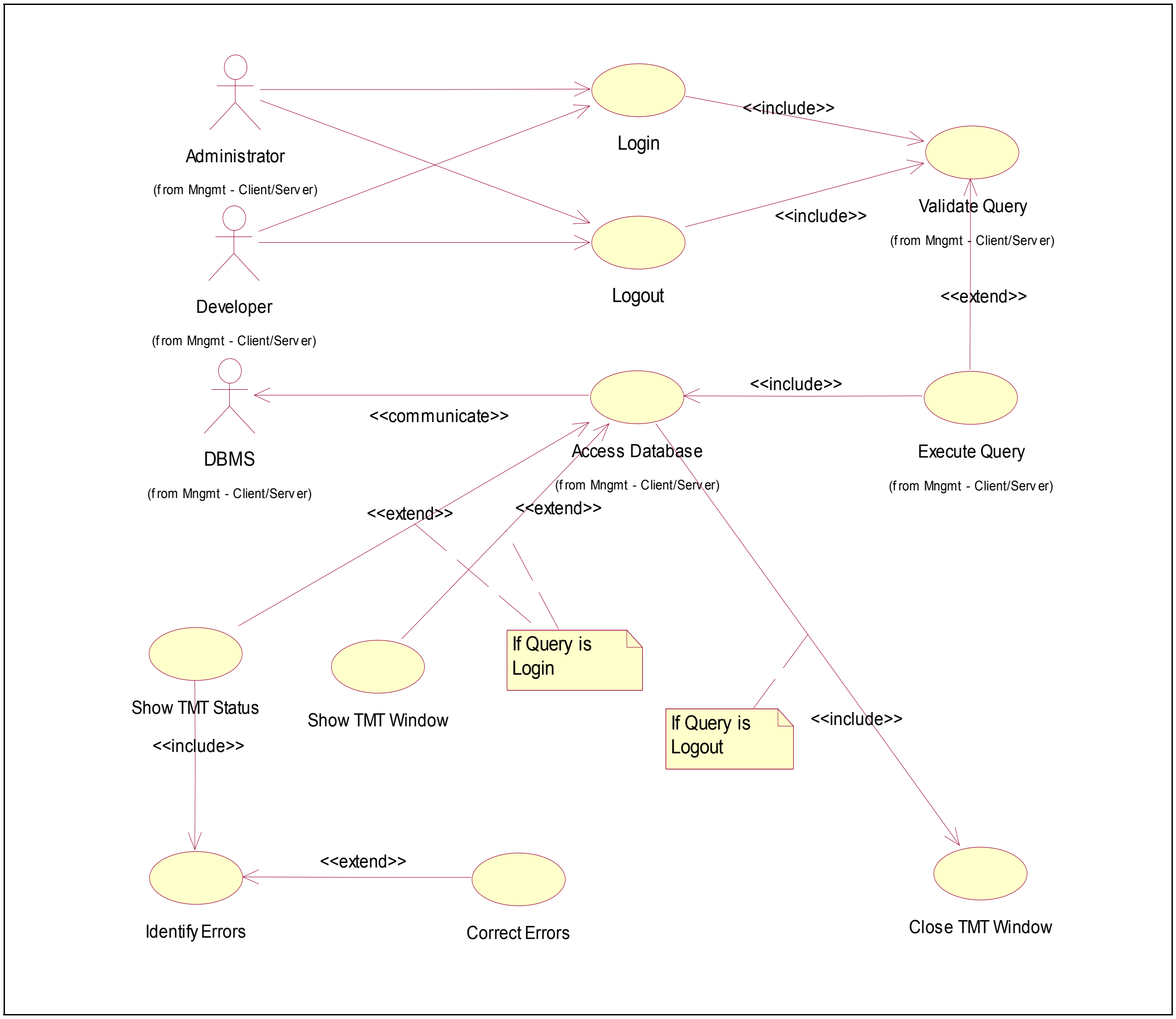


Figure 5 Login/Logout Mngmt

4. Login

4.1 Brief Description

Allowing all user types to access their respective client module depending on the username and password entered.

4.2 Flow of Events

4.2.1 Basic Flow

The user provides the username and password strings to the Login interface and activates the login process. Entries are validated through a database query and results are returned to the user.

4.2.2 Alternative

4.2.2.1 <User is *Developer*>

Username and password match a developer entry in the database, DCM loading is launched.

4.2.2.2 <User is *Administrator*>

Username and password match an administrator entry in the database, MCM loading is launched.

4.3 Preconditions

4.3.1 <Predefined user profile>

The *Administrator* must have, previously, defined a user account.

4.3.2 <Possession of user login information>

All users must have in hand their respective username and password.

4.4 Postconditions

4.4.1 <Server Module Response>

The Login Query cannot be terminated before the Server Module sends its response to the Client Module.

4.5 Extension Points

4.5.1 <Validate Query - include>

The application validates the entered information: username and password

4.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

4.7 Other Diagrams

None.

5. Logout

5.1 Brief Description

Allow users to terminate a client session

5.2 Flow of Events

5.2.1 Basic Flow

Users send the logout query to the application through their respective GUI. The system disconnects and logs out the user.

5.2.2 Alternative

None.

5.3 Preconditions

None

5.4 Postconditions

5.4.1 <Server Module Response>

The Logout Query cannot be terminated before the Server Module sends its response to the Client Module.

5.5 Extension Points

5.5.1 <Validate Query - include>

The application validates the entered information: username and password

5.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

5.7 Other Diagrams

None.

6. Show TMT Status

6.1 Brief Description

Any error of execution, communication, validation or else is identified during an opened session. If the Server Module cannot correct the identified error, an appropriate comment is displayed.

6.2 Flow of Events

6.2.1 Basic Flow

At any time, the Server Module verifies the TMT status and identifies errors, if any. Errors are shown to the users using a pop-up window with a text-based comment or the standard HTTP error messages pages.

6.2.2 Alternative

None.

6.3 Preconditions

None.

6.4 Postconditions

None.

6.5 Extension Points

6.5.1 <Identify errors - include>

If any error is present, the Server Module identifies the error specific information.

6.5.2 <Correct errors - extend>

If the error is known and can be managed, the Server Module corrects it without showing any message to the user. This process is user-transparent.

6.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

6.7 Other Diagrams

None.

7. Identify errors

7.1 Brief Description

The Server Module identifies all possible errors (execution, communication, validation or else) upon managing all requested operations. If possible, the Server Module autocorrects the error else it calls the appropriate dialog box and show the error to the user.

7.2 Flow of Events

7.2.1 Basic Flow

The server module identifies the errors, if any.

7.2.2 Alternative

None.

7.3 Preconditions

None.

7.4 Postconditions

None.

7.5 Extension Points

7.5.1 <Correct errors - extend>

The server module corrects the error, if possible.

7.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

7.7 Other Diagrams

None.

8. Correct errors

8.1 Brief Description

Upon identifying system errors, the Server Module determines if that error is fatal or if abnormal termination can be avoided by auto correction. If yes, the Server Module corrects the error and the process remains transparent to the user.

8.2 Flow of Events

8.2.1 Basic Flow

The non-critical error has already been identified. Then, the Server Module corrects the error.

8.2.2 Alternative

None.

8.3 Preconditions

8.3.1 <Non-Critical Error Presence>

The Server Module can autocorrect all non-critical errors. This use-case will not occur if no error has been identified.

8.4 Postconditions

8.4.1 <First Postcondition>

The application runs normally.

8.5 Extension Points

None.

8.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

8.7 Other Diagrams

None.

9. Show TMT Window

9.1 Brief Description

If either a Developer or an Administrator is using the TMT, the right window must be loaded. Thus, when the DCM is loaded, the TMT Window for Developer is also generated by the Browser. If the MCM is loaded, the TMT Window for Administrator is also generated by the Browser.

9.2 Flow of Events

9.2.1 Basic Flow

Depending on what username and password have been entered, the Browser ensures that the DCM or MCM are loaded (Load DCM or Load MCM) correctly and then loads the corresponding GUI to the *Developer or Administrator*. All GUI components are members of the TMT Window.

9.2.2 Alternative

9.2.2.1 <Developer Window>

If the user is a Developer, it loads the Developer GUI.

9.2.2.2 <Administrator Window>

If the user is a Developer, it loads the Administrator GUI.

9.3 Preconditions

None.

9.4 Postconditions

None.

9.5 Extension Points

None.

9.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

9.7 Other Diagrams

None.

10. Close TMT Window

10.1 Brief Description

On a logout query, the TMT Window closes.

10.2 Flow of Events

10.2.1 Basic Flow

The user requires a Logout, the query is validated and the TMT Window is closed, closing also all opened modules (SM, DCM or MCM)

10.2.2 Alternative

None.

10.3 Preconditions

10.3.1 <Logout query>

A Logout query must have been launched.

10.4 Postconditions

None.

10.5 Extension Points

None.

10.6 Use-Case Diagrams

Refer to the Section Use-Case Diagram (Figure 5)

10.7 Other Diagrams

None.

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
User Interface Prototype

Version <3.0>

Revision History

Date	Version	Description	Author
15/02/2001	01	First version	Sandra Lee
20/02/2001	02	Modifications applied as requested by lab attendant.	Sandra Lee
02/04/2001	03	Major updates of all GUI images.	James Prevost

Preface

The following case study has been modified from its original content. The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Requirements Section.

Regards,

Unified Process for Education Team

Table of Content

Figures

User Interface Prototype

1. Introduction

1.1 Purpose

The User Interface Prototype document presents explicitly what the TMT user will see on the screen and how to use these different graphical interfaces. This document represents also, graphically, the model developed and detailed in the project's documentation. Moreover, this document can be used as a demonstration for future end-users in order to preview the final product.

1.2 Scope

The following roles use the user-interface prototype:

- Analysts, to understand the user interface for a use case and how the user interface impacts the analysis of the system
- Designers, to understand how the user interface impacts and what it requires from the "inside" of the system
- Testers, to plan testing activities.
- Stakeholders as a preview of the interface they will use.

1.3 Definitions, Acronyms, and Abbreviations

Refer to the project's Glossary. See References.

1.4 References

- Glossary Document, Glossary, TMT Team 1, École Polytechnique de Montréal, 2001.
- [Course Web Page INF4302 : http ://www.cours.polymtl.ca/inf4302/
- UPEDU : http://www.upedu.org

1.5 Overview

This document is divided in three main sections: Identification interfaces, Developer Client-Module Interface and Manager Client-Module Interface. All sections contain graphical information about their related primary and secondary windows.

Further you will find a brief description on the different graphics standards used in the Time Monitoring Tool and an interface browsing hierarchy.

2. Identification Interfaces

2.1 Primary Windows

2.1.1 <User Identification>

2.1.1.1 Purpose

When a user wishes to use the TMT, he has to identify himself with a username and password. The user identification window ensures that all fields (username and password) are filled correctly. The validation is made upon clicking the « Enter » button.

2.1.1.2 Objects and actions

Objects	Actions
Text Input – Username	Allows the user to type in his username
Text Input – Password	Allows the user to type in his password
Button – Enter	Launch the username/password validation query and grant the user access if provided information is validated. Otherwise redirects the user to an error page.
Button – Reset	Allows the user to erase the content of the two Text Input field described above

2.1.1.3 Composites

This window contains 5 frames (web frames):

- **Left Frame: Acts as a border frame, containing the gray background the left side.**
- **Right Frame: Acts as a border frame, containing the gray background the right side.**
- **Top Frame: Acts as a border frame, containing the gray background the top side.**
- **Bottom Frame: Acts as a border frame, containing the gray background the bottom side.**
- **Main Frame: Displays the information and action objects of the <*User Identification*> primary window**

2.1.1.4 Visual Dimensions

Position

This window has the following attributes: *scrollable, movable* and all user-defined position properties of the browser used by the user (Standard web window)

Size

This window has the following attributes: *minimizable, restorable, resizable, closeable* and all user-defined size properties of the browser used by the user (Standard web window)

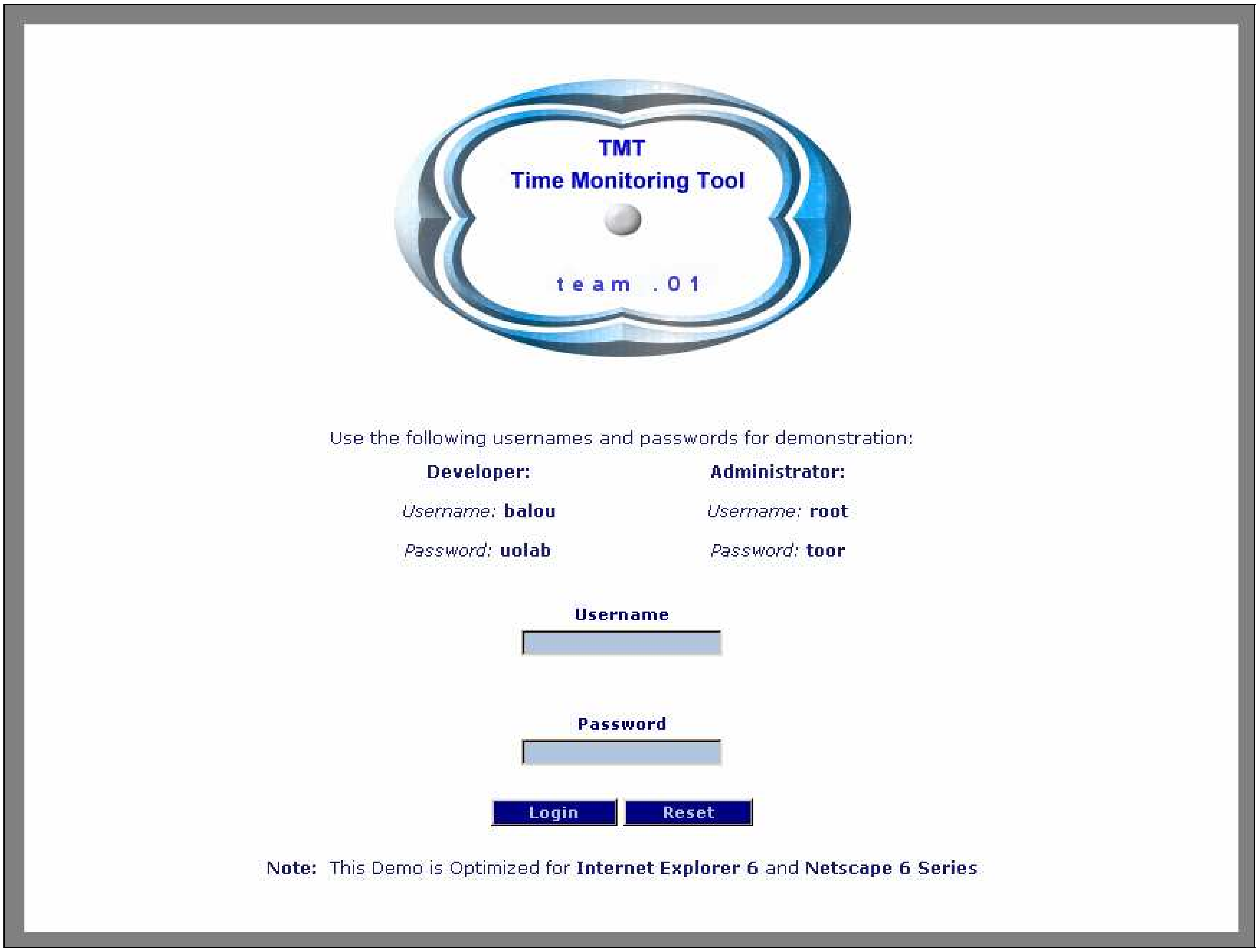
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

2.1.1.5 Screenshot Sample



Use the following usernames and passwords for demonstration:

Developer:	Administrator:
Username: **balou**	*Username:* **root**
Password: **uolab**	*Password:* **toor**

Note: This Demo is Optimized for **Internet Explorer 6** and **Netscape 6 Series**

Figure 1: User Identification Interface

2.2 Secondary Windows: Property Windows

None.

2.3 Secondary Windows: Other Windows

2.3.1 <Wrong User Name>

2.3.1.1 Purpose

If the entered username doesn't exist in the Database, the validation process redirects the user to a Wrong Username page.

2.3.1.2 Objects and actions

Objects	Actions
Hyperlink – Try Again	**Allows the user to go back to the primary window and type in his username and password again**

2.3.1.3 Composites

None

2.3.1.4 Visual Dimensions

Position

This window has the following attributes: ***floating (Top left of screen)***

Size

This window has the following attributes: ***minimizable, closeable***

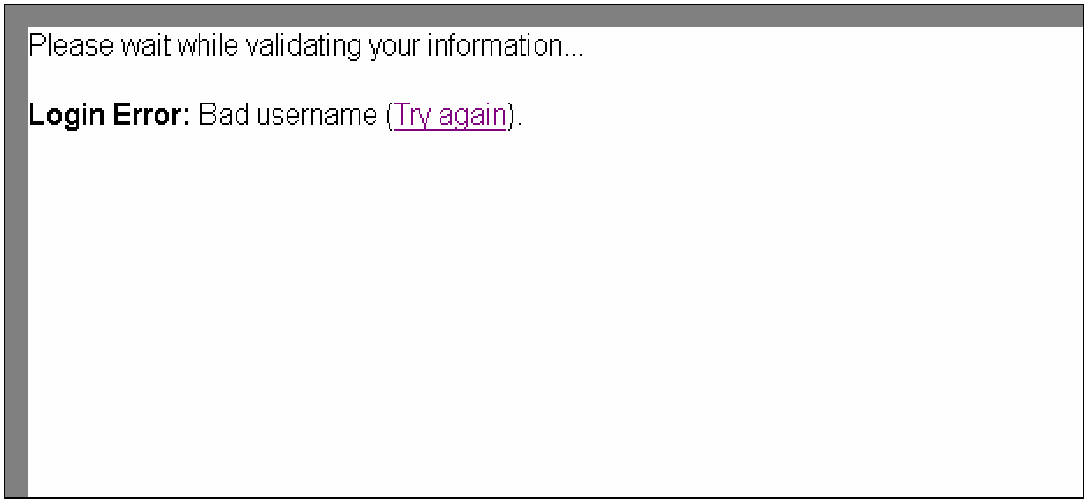
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

2.3.1.5 Screenshot Sample



Please wait while validating your information...

Login Error: Bad username (Try again).

Figure 2: Wrong Username Interface

2.3.2 <Wrong Password>

2.3.2.1 Purpose

Once the username has been validated, if the entered password doesn't match the username in the Database, the validation process redirects the user to a Wrong Password page.

2.3.2.2 Objects and actions

Objects	Actions
Hyperlink – Try Again	**Allows the user to go back to the primary window and type in his username and password again**

2.3.2.3 Composites

None

2.3.2.4 Visual Dimensions

Position

This window has the following attributes: *floating (Top left of screen)*

Size

This window has the following attributes: *minimizable, closeable*

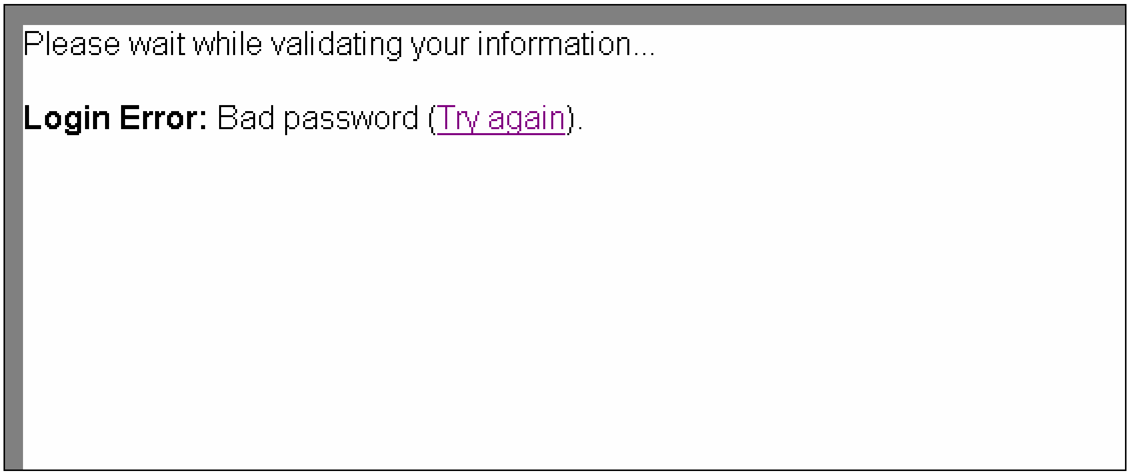
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

2.3.2.5 Screenshot Sample



Please wait while validating your information...

Login Error: Bad password (Try again).

Figure 3: Wrong Password Interface

2.3.3 <Account Deactivated>

2.3.3.1 Purpose

Once the username and password have been validated, the user profile is deactivated; the validation process redirects the user to an Account Deactivated Page.

2.3.3.2 Objects and actions

Objects	Actions
Hyperlink – Back to Login Screen	Allows the user to go back to the primary window

2.3.3.3 Composites

None

2.3.3.4 Visual Dimensions

Position

This window has the following attributes: *floating (Top left of screen)*

Size

This window has the following attributes: *minimizable, closeable*

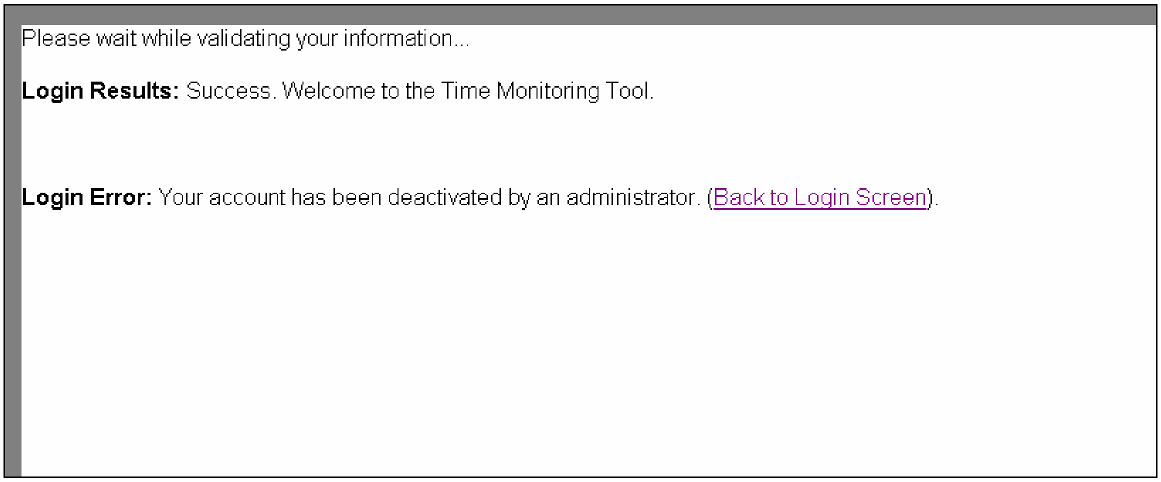
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

2.3.3.5 Screenshot Sample



Please wait while validating your information...

Login Results: Success. Welcome to the Time Monitoring Tool.

Login Error: Your account has been deactivated by an administrator. (Back to Login Screen).

Figure 4: Account Deactivated Interface

3. Developer Client-Module Interfaces

3.1 Primary Windows

3.1.1 <Developer Client module Main Screen>

3.1.1.1 Purpose

If the login step has been validated, a Developer gets the following interface. This is his main work screen where he can view, modify or add timestamps. Moreover, this primary window shows some timestamps statistics (Total hours stamped).

3.1.1.2 Objects and actions

Objects	Actions
Text Input – Username	Display only.
Text Input – Date/Time	Display only
Button – CheckAll Timestamps to resubmit (TR)	Enables all check boxes in the TR table.
Check Box – TR	Enables one row in the TR table
Text Input – Date TR	Date of submitted TR
Select Box – Project TR	Project of TR
Text Input – Start TR	Starting hour of TR
Text Input – End TR	Ending hour of TR
Text Input – Length TR	Length of TR
Select Box – Task TR	Task name TR
Select Box – Artifact TR	Artifact name TR
Select Box – Activity TR	Related activity name TR
Text Input – Description TR	Brief task description TR
Button - Resubmit timestamps	Allows the developer to re-submit his timestamp
Text Input – Date New Timestamp (NT)	Date of NT
JScript Button – Calendar NT	Pops up a calendar JavaScript Window allowing the developer to select a specific date instead of type manually the date in the timestamp date field.
Select Box – Project NT	Project of NT
Text Input – Start NT	Starting hour of NT
Text Input – End NT	Ending hour of NT

Text Input – Length NT	**Length of NT**
Select Box – Task NT	**Task name NT**
Select Box – Artifact NT	**Artifact name NT**
Select Box – Activity NT	**Related activity name NT**
Text Input – Description NT	**Brief task description NT**
Button – Submit NT	**Allows the developer to submit his timestamp**
Button – End session	**Terminates the developer session and logs out**
Button – View Stats	**Shows the timestamps current week's statistics**

3.1.1.3 Composites

This window contains 5 frames (web frames):

- **Left Frame: Acts as a border frame, containing the gray background the left side.**
- **Right Frame: Acts as a border frame, containing the gray background the right side.**
- **Top Frame: Acts as a border frame, containing the gray background the top side.**
- **Bottom Frame: Acts as a border frame, containing the gray background the bottom side.**
- **Main Frame: Displays the information and action objects of the <** ***Developer Client module Main Screen*** **> primary window**

This window contains 2dynamic tables driven with JSP technology and static HTML table.

Dynamic tables:

- ***Current week timestamps; Included*** **components are: indicator (non-validated and validated timestamps), date, project, starting hour, ending hour, length, task, artifact, activity and description. This is a read-only table.**

- ***Timestamps to be re-submitted; Included*** **components are: CheckAll box, specific timestamp checkbox, date, project, starting hour, ending hour, length, task, artifact, activity and description.**

Static HTML table:

- ***New Timestamps to be submitted; Included*** **components are: date, JScript Calendar button, project, starting hour, ending hour, length, task, artifact, activity and description. This is a read-only table. There are a total of 5 rows in that table.**

3.1.1.4 Visual Dimensions

Position

This window has the following attributes: ***scrollable, movable*** **and all user-defined position properties of the browser used by the user (Standard web window)**

Size

This window has the following attributes: ***minimizable, restorable, resizable, closeable*** and all user-defined size properties of the browser used by the user (Standard web window)

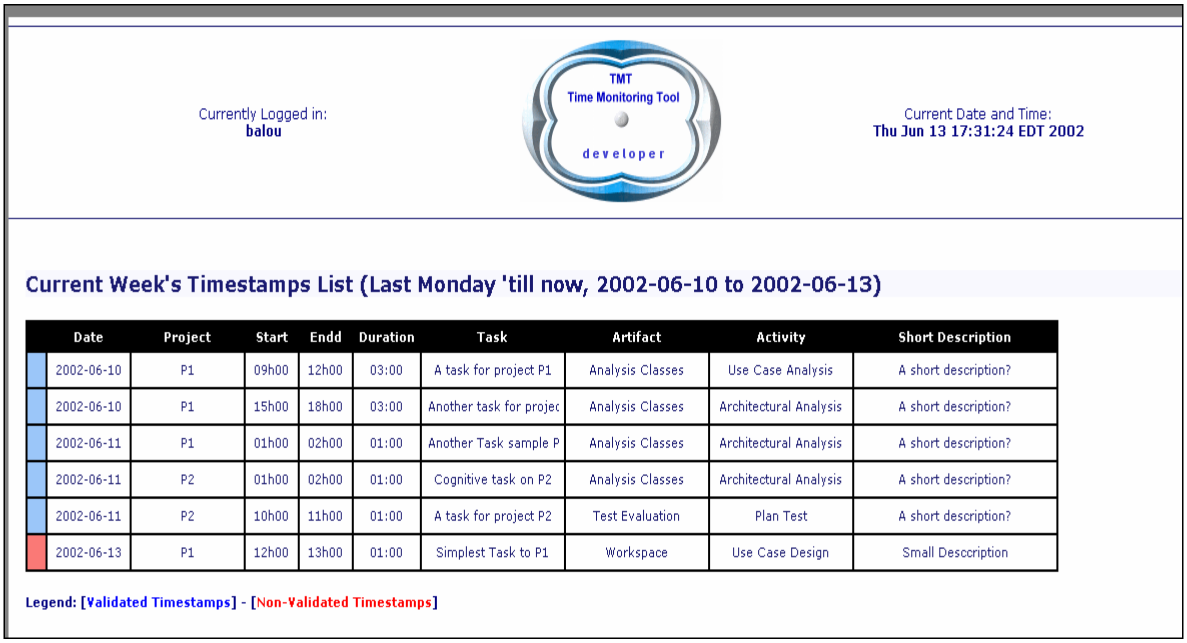
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

3.1.1.5 Screenshot Sample



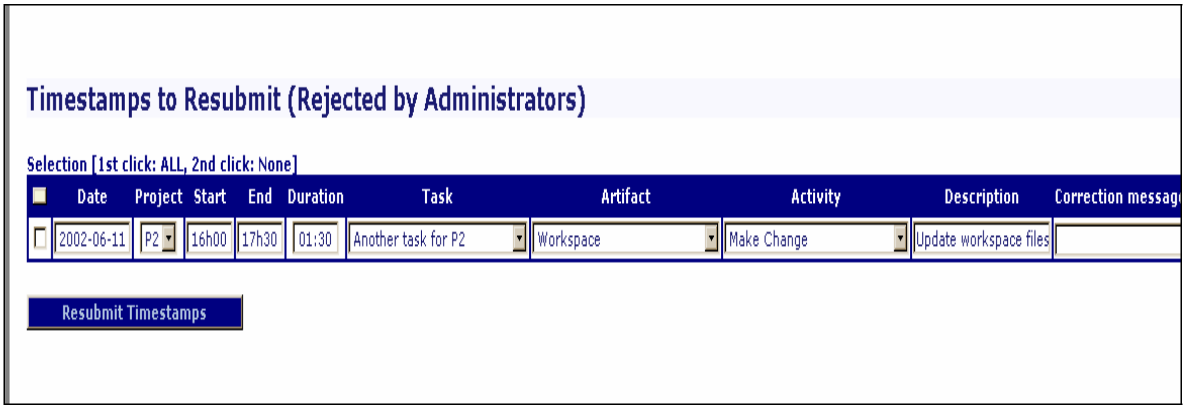
Currently Logged in:
balou

Current Date and Time:
Thu Jun 13 17:31:24 EDT 2002

Current Week's Timestamps List (Last Monday 'till now, 2002-06-10 to 2002-06-13)

	Date	Project	Start	Endd	Duration	Task	Artifact	Activity	Short Description
🟦	2002-06-10	P1	09h00	12h00	03:00	A task for project P1	Analysis Classes	Use Case Analysis	A short description?
🟦	2002-06-10	P1	15h00	18h00	03:00	Another task for projec	Analysis Classes	Architectural Analysis	A short description?
🟦	2002-06-11	P1	01h00	02h00	01:00	Another Task sample P	Analysis Classes	Architectural Analysis	A short description?
🟦	2002-06-11	P2	01h00	02h00	01:00	Cognitive task on P2	Analysis Classes	Architectural Analysis	A short description?
🟦	2002-06-11	P2	10h00	11h00	01:00	A task for project P2	Test Evaluation	Plan Test	A short description?
🟥	2002-06-13	P1	12h00	13h00	01:00	Simplest Task to P1	Workspace	Use Case Design	Small Desccription

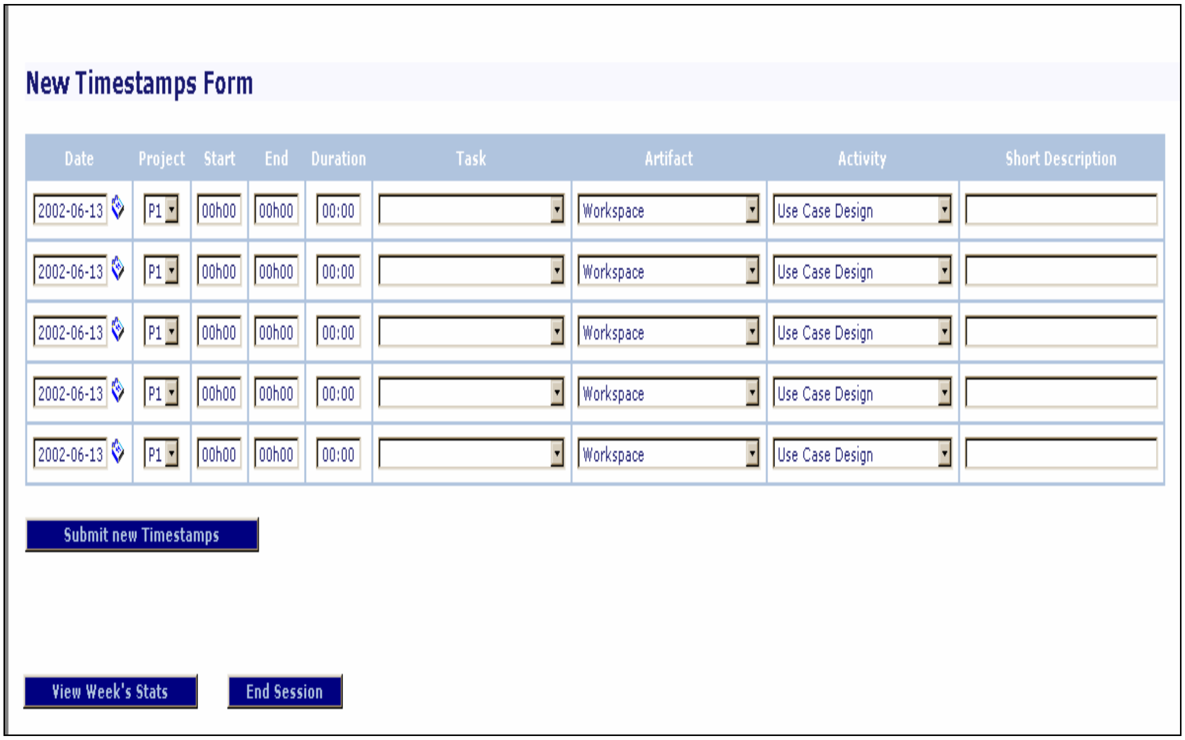
Legend: [Validated Timestamps] - [Non-Validated Timestamps]



Timestamps to Resubmit (Rejected by Administrators)

Selection [1st click: ALL, 2nd click: None]

□	Date	Project	Start	End	Duration	Task	Artifact	Activity	Description	Correction messag
□	2002-06-11	P2	16h00	17h30	01:30	Another task for P2	Workspace	Make Change	Update workspace files	



New Timestamps Form

Date	Project	Start	End	Duration	Task	Artifact	Activity	Short Description
2002-06-13	P1	00h00	00h00	00:00		Workspace	Use Case Design	
2002-06-13	P1	00h00	00h00	00:00		Workspace	Use Case Design	
2002-06-13	P1	00h00	00h00	00:00		Workspace	Use Case Design	
2002-06-13	P1	00h00	00h00	00:00		Workspace	Use Case Design	
2002-06-13	P1	00h00	00h00	00:00		Workspace	Use Case Design	

Figure 5: Developer Client module Main Screen Interface (3 main parts) - Timestamps

3.2 Secondary Windows: Property Windows

None.

3.3 Secondary Windows: Other Windows

3.3.1 <Timestamps Statistics>

3.3.1.1 Purpose

Shows weekly statistics for the current user. Timestamps durations are calculated by day, by project and by task.

3.3.1.2 Objects and actions

Objects	Actions
Button – Back to the DCM	**Allows the user to go back to the primary window and view the DCM data**

3.3.1.3 Composites

None

3.3.1.4 Visual Dimensions

Position

This window has the following attributes: *scrollable, movable* and all user-defined position properties of the browser used by the user (Standard web window)

Size

This window has the following attributes: *minimizable, restorable, resizable, closeable* and all user-defined size properties of the browser used by the user (Standard web window)

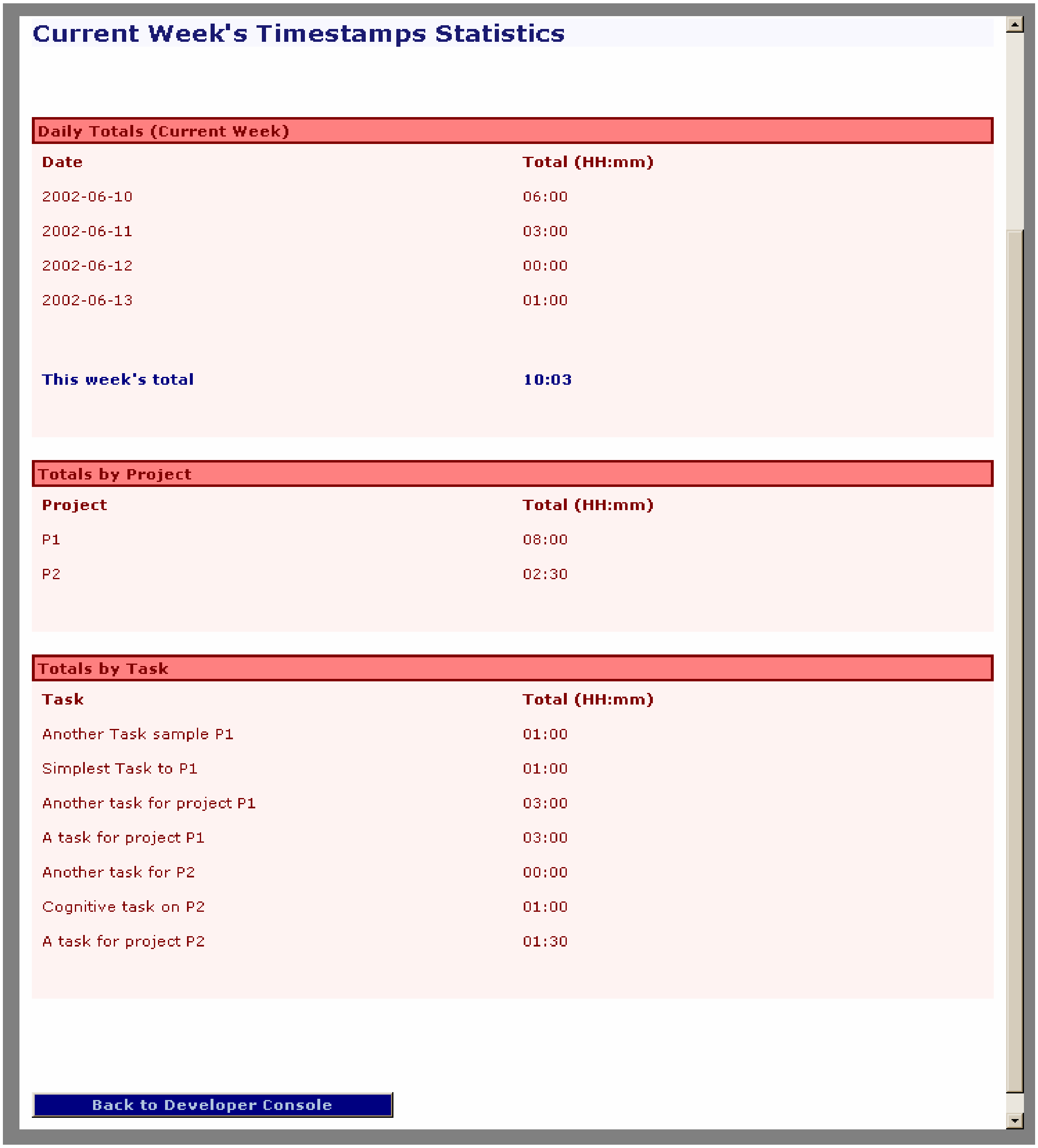
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

3.3.1.5 Screenshot Sample



Current Week's Timestamps Statistics

Daily Totals (Current Week)

Date	Total (HH:mm)
2002-06-10	06:00
2002-06-11	03:00
2002-06-12	00:00
2002-06-13	01:00
This week's total	10:03

Totals by Project

Project	Total (HH:mm)
P1	08:00
P2	02:30

Totals by Task

Task	Total (HH:mm)
Another Task sample P1	01:00
Simplest Task to P1	01:00
Another task for project P1	03:00
A task for project P1	03:00
Another task for P2	00:00
Cognitive task on P2	01:00
A task for project P2	01:30

Figure 6: Timestamps Statistics Interface

4. Manager Client-Module Interfaces

4.1 Primary Windows

4.1.1 <Manager Client-Module Main Screen>

4.1.1.1 Purpose

If the login step has been validated, a Manager gets the following interface. This is his main work screen where he can manage system users (Add, list, deactivate, modify, validate timestamps) or manage active projects (add project, add task, Export to MS-Excel, Export to MS-Project).

4.1.1.2 Objects and actions

Objects	Actions
Text Input – Username	**Display only.**
Text Input – Date/Time	**Display only**
JScript Object – Left Menu Option: Add user	**Redirects the manager to the *Add a new user Secondary Window***
JScript Object – Left Menu Option: Validate timestamps	**Redirects the manager to the *Validate New Timestamps Secondary Window***
JScript Object – Left Menu Option: List/Modify/Deactivate	**Redirects the manager to the *List/Modify/Deactivate Secondary Window***
JScript Object – Right Menu Option: Add a project	**Redirects the manager to the *Add a Project Secondary Window***
JScript Object – Right Menu Option: Add a task	**Redirects the manager to the *Add a Task Secondary Window***
JScript Object – Right Menu Option: Export to MSEXCEL	**Redirects the manager to the *Export to EXCEL Secondary Window***
JScript Object – Right Menu Option: Export to MSPROJECT	**Redirects the manager to the *Export to MSPROJECT Secondary Window***

4.1.1.3 Composites

This window contains 5 frames (web frames):

- **Left Frame: Acts as a border frame, containing the gray background the left side.**
- **Right Frame: Acts as a border frame, containing the gray background the right side.**
- **Top Frame: Acts as a border frame, containing the gray background the top side.**
- **Bottom Frame: Acts as a border frame, containing the gray background the bottom side.**
- **Main Frame: Displays the information and action objects of the < *Manager Client-Module Main Screen* > primary window**

This window contains 2 JavaScript Menu Objects: The left menu has 3 submenu objects and the right menu has 4 submenu objects.

4.1.1.4 Visual Dimensions

Position

This window has the following attributes: *scrollable, movable* and all user-defined position properties of the browser used by the user (Standard web window)

Size

This window has the following attributes: *minimizable, restorable, resizable, closeable* and all user-defined size properties of the browser used by the user (Standard web window)

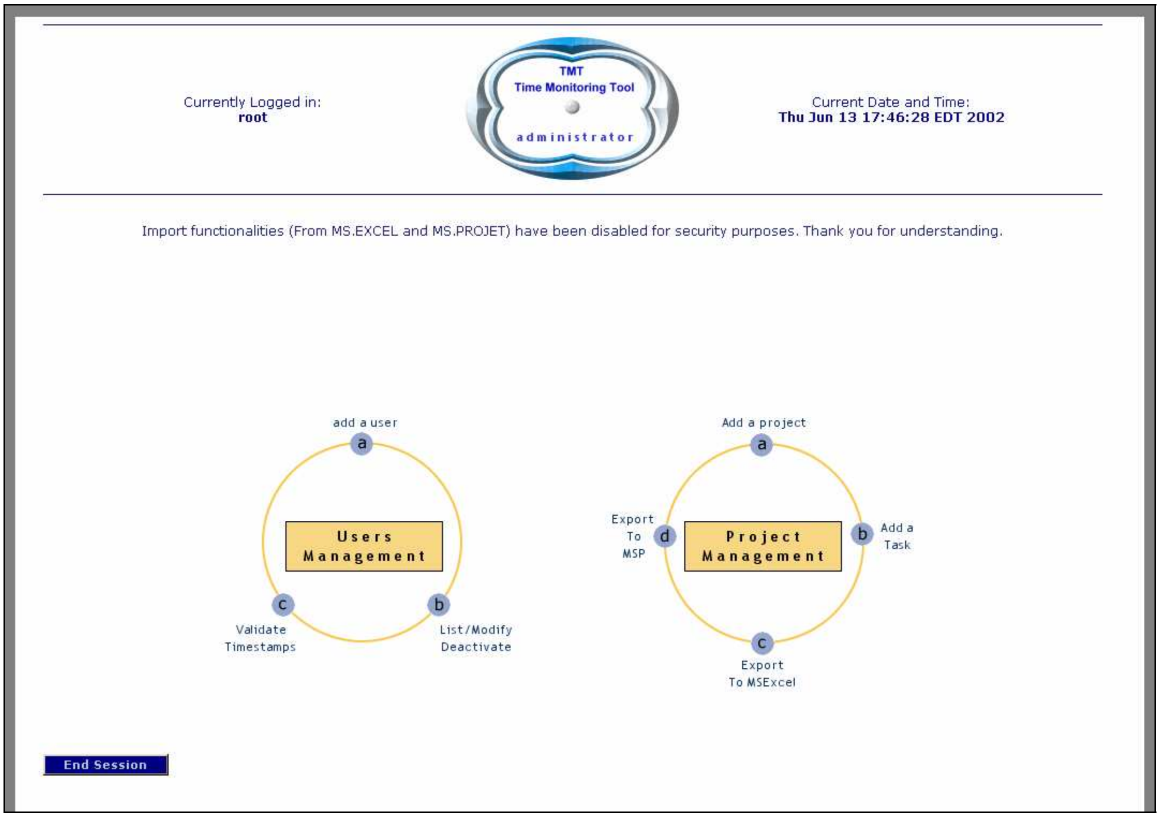
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

4.1.1.5 Screenshot Sample



Currently Logged in:
root

Current Date and Time:
Thu Jun 13 17:46:28 EDT 2002

Import functionalities (From MS.EXCEL and MS.PROJET) have been disabled for security purposes. Thank you for understanding.

Figure 7: Manager Client-Module Main Screen Interface

4.2 Secondary Windows: Property Windows

4.2.1 <Add a new user>

4.2.1.1 Purpose

The Administrator can add users (developers) through this interface. Required fields are username, password, superior, and timestamps relative information. A submit button is use to activate the database INSERT query.

4.2.1.2 Objects and actions

Objects	Actions
Text Input – Username	Display only.
Text Input – Date/Time	Display only
Text Input – Username	Allows the manager to enter the developer's name
Text Input – Password	Allows the manager to enter the developer's password
Select Box – Superior	Allows the manager to select the developer's superior
Radio Buttons – Status	Allows the manager to select if the new user is a developer or a manager
Multiple Select Box - Projects	Allows the manager to multiple-select projects for the developer
Select Box – Default Project	Allows the manager to select a default project for the developer
Select Box – Default Task	Allows the manager to select a default task for the developer
Select Box – Default Activity	Allows the manager to select a default activity for the developer
Select Box – Default Artifact	Allows the manager to select a default artifact for the developer
Button – Submit	Submits the form and performs the Database INSERT query
Button – Reset/Erase	Erases all fields in the form
Hyperlink – Back to MCM	Redirects the manager to the main screen (The menus)

4.2.1.3 Composites

This window contains 5 frames (web frames):

- **Left Frame: Acts as a border frame, containing the gray background the left side.**
- **Right Frame: Acts as a border frame, containing the gray background the right side.**
- **Top Frame: Acts as a border frame, containing the gray background the top side.**
- **Bottom Frame: Acts as a border frame, containing the gray background the bottom side.**
- **Main Frame: Displays the information and action objects of the <*User Identification*> primary window**

This window contains 1 static HTML table containing all fields object described in the previous section.

4.2.1.4 Visual Dimensions

Position

This window has the following attributes: *scrollable, movable* and all user-defined position properties of the browser used by the user (Standard web window)

Size

This window has the following attributes: *minimizable, restorable, resizable, closeable* and all user-defined size properties of the browser used by the user (Standard web window)

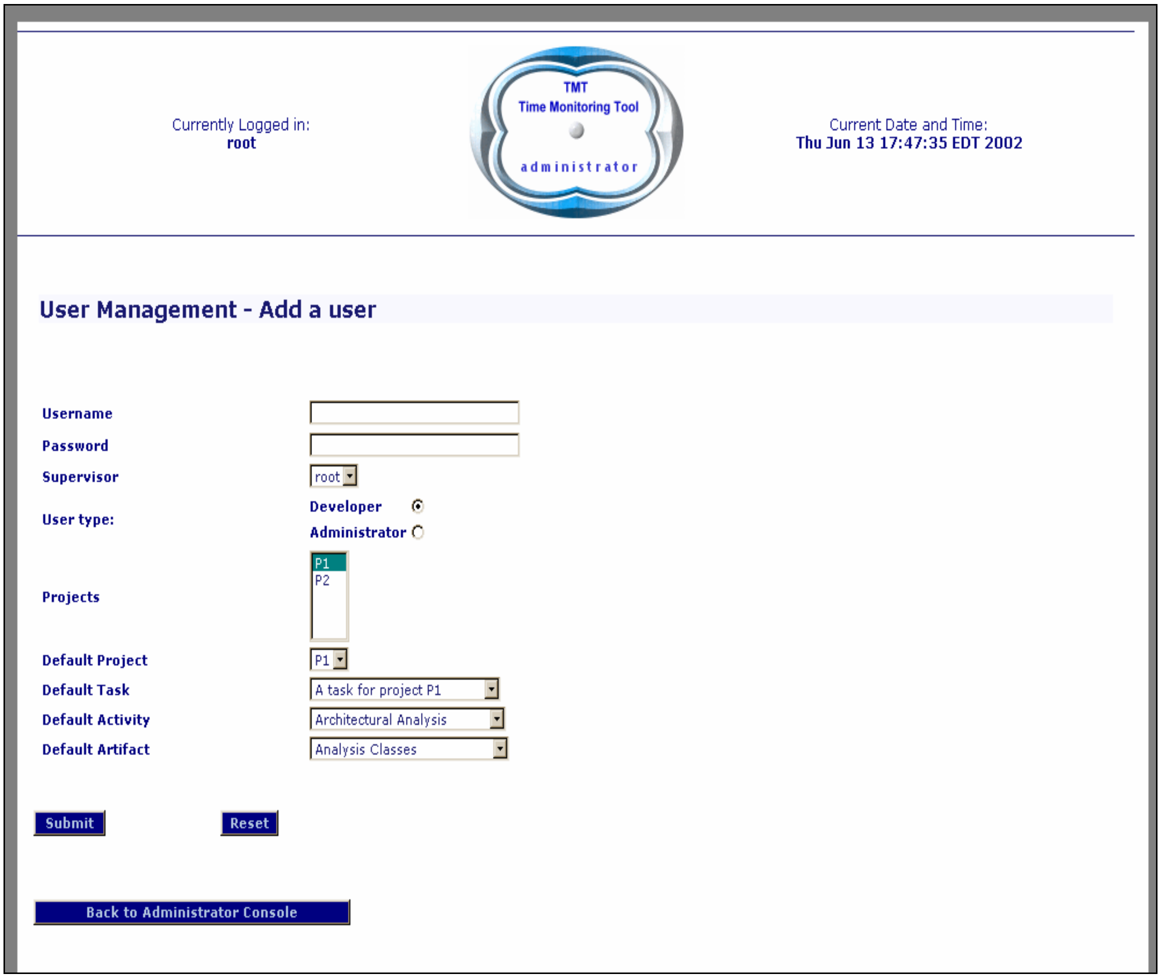
Shape

Standard web window (Rectangle).

Color

No specific color schemes defined.

4.2.1.5 Screenshot Sample



Currently Logged in:
root

Current Date and Time:
Thu Jun 13 17:47:35 EDT 2002

User Management - Add a user

Username	
Password	
Supervisor	root
User type:	Developer ● Administrator ○
Projects	P1 P2
Default Project	P1
Default Task	A task for project P1
Default Activity	Architectural Analysis
Default Artifact	Analysis Classes

Figure 8: Add a new user

4.2.2 <Validate new timestamps>

[...]

4.3 Secondary Windows: Other Windows

None

5. Graphics Standards

All graphical interfaces and elements use HTML, DHTML and JavaScript web technologies interlaced with standard images formats (.GIF, .JPG, .PNG).

The color schemes may require a True Color Compatible graphical adapter and at least an 800x600 screen resolution. Best viewed at 1024x768 and over.

Al fonts used are in the Helvetica and Arial font-family.

All graphical interfaces are best viewed in Internet Explorer 5+ and Netscape Navigator 4.79+

6. Other Standards

The TMT logo used has been designed by TMT Team 4 and supervised by member John Lemon.

If any special colors are used for a specific purpose, the explanation can be found on the web page itself (Legend or note)

7. Browsing Hierarchy

The following is a window hierarchy diagram showing how the different windows are linked together.

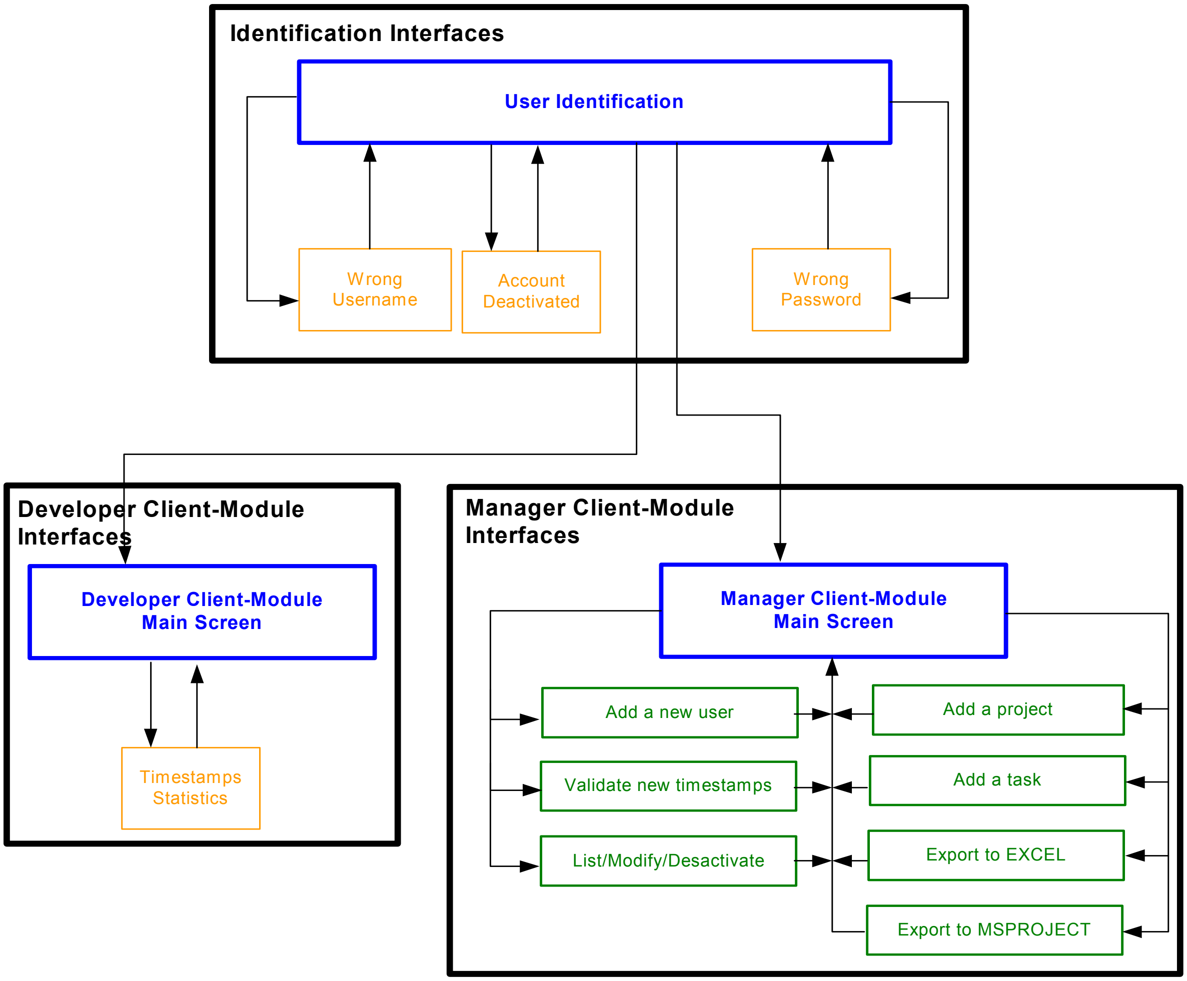


Figure 9: Graphical Interfaces Hierarchy

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
Software Architecture Document

**Version <1.0>

Revision History

Date	Version	Description	Author
26/01/01	<1.0>	First Version	Robert Latour

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This Software Architecture Document provides an architectural overview of the Time Monitoring Tool. The Time Monitoring Tool allows developers working within a defined software development process to record the time spent on the various activities, in a database. The TMT will also allow an Administrator to derive analyses and produce reports based on the data entered in the system.

1.3 Definitions, Acronyms, and Abbreviations

See Glossary, document upedu_ex_gloss.pdf

1.4 References

1. TMT - Glossary
2. TMT - Use Case Specification
3. TMT - Supplementary Specification
4. TMT - Software Development Plan
5. TMT - Iteration Plan

2. Architectural Representation

This document presents the architectural as a series of views; use case view, process view, deployment view, and implementation view. These views are presented as Rational Rose Models and use the Unified Modeling Language (UML).

3. Architectural Goals and Constraints

The TMT system to be developed is a stand-alone tool that is integrated within the organization's Intranet. It consists in four major components: a Developer Client Module, a Server Module, a Database, and an Administrator Client Module

All components must execute on a WindowsNT personal computer.

The Developer and Administrator Client Modules must communicate with the server over a TCP/IP connection.

The Server and the Database components should be located on the same host.

4. Use-Case View

The Use Case View is important input to the selection of the set of scenarios and/or use cases that are the focus of an iteration. It describes the set of scenarios and/or use cases that represent some significant, central functionality. It also describes the set of scenarios and/or use cases that have a substantial architectural coverage (that exercise many architectural elements) or that stress or illustrate a specific, delicate point of the architecture.

Refer to *Use-Case Specifications document* for more information- upedu_ex_ucspec.pdf

4.1 Use-Case Realizations

Refer *Use Case Realization document* – upedu_ex_ucrea.pdf

5. Logical View

This section describes the architecturally significant parts of the design model, such as its decomposition into subsystems and packages. And for each significant package, its decomposition into classes and class utilities.

5.1 Overview

This subsection describes visually the overall decomposition of the design model in terms of its package hierarchy and layers.

5.2 Architecturally Significant Design Packages

5.2.1 Design Model: Packages Diagrams

The design model represents explicitly the structure and organization of the TMT system. Packages and corresponding classes are presented with a brief description.

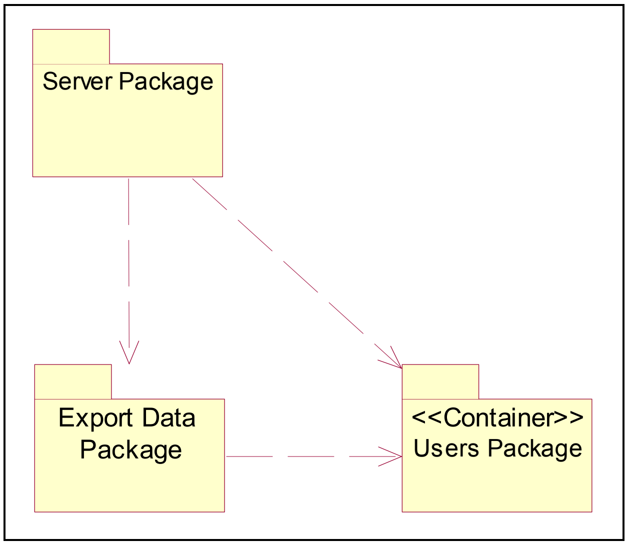


Figure 1: Design Model Packages Level 1

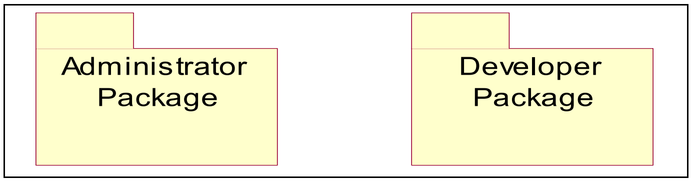


Figure 2: Design Model Packages Level 2 (Users Packages)

5.2.2 Packages Description

5.2.2.1 Level 1 Packages

Server	
Description :	Main System Package. Although this package is dependant of other system packages, this package is the central point of all decisions made in the system. All client queries are managed by this package.
Corresponding classes* :	Observable Observer AdministratorObserver DeveloperObserver TMTServer
Relations :	Main TMT package. Dependant of: Developer, Administrator and Export Data packages
Sub packages :	Users, Export Data

Users	
Description :	Container Package for the Developer and Administrator Modules
Relations :	Is a sub package of the main package Server.
Sub packages :	None.

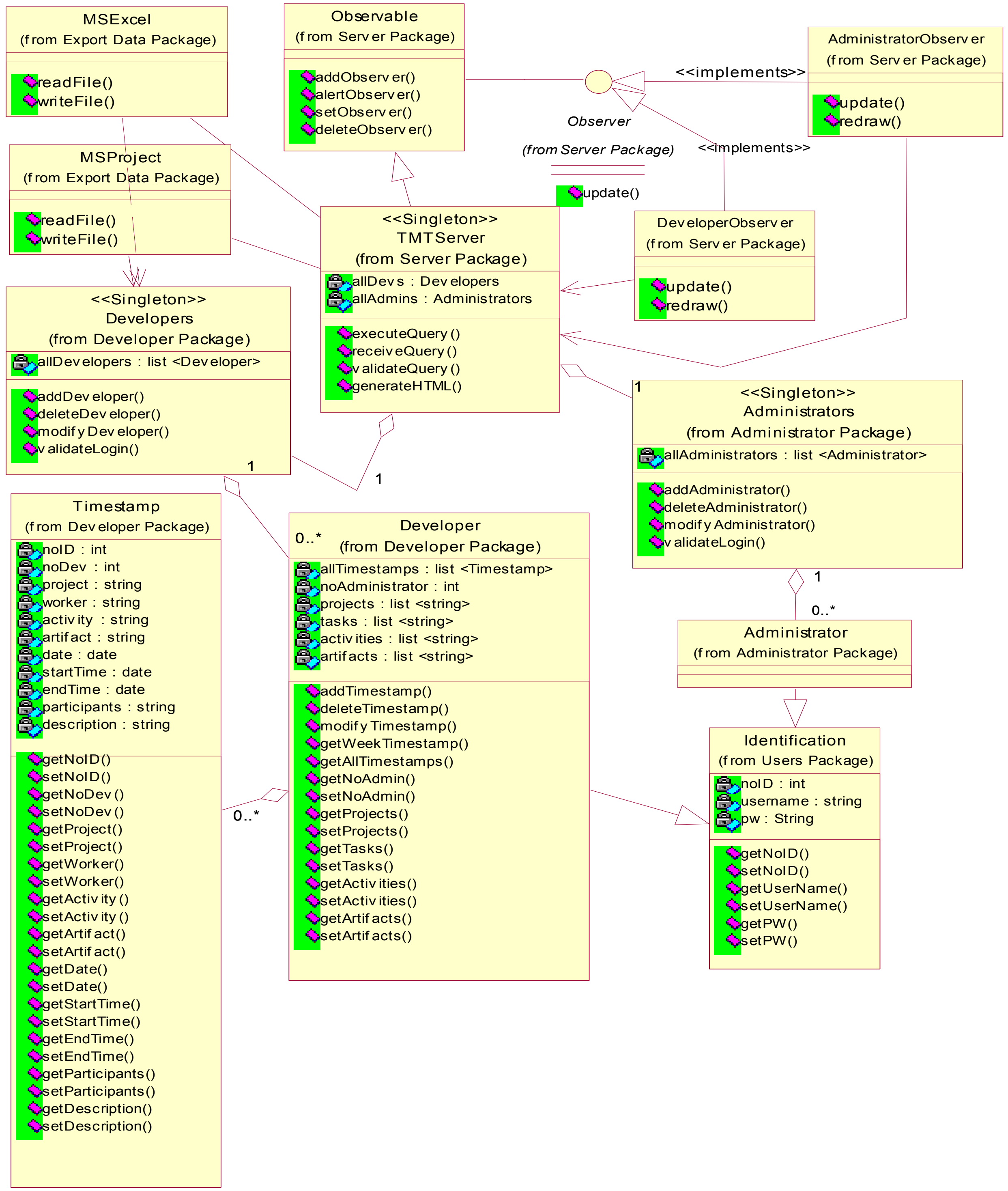
ExportData	
Description :	This package includes all necessary methods to import or export from/to Excel and MsProject. This package is dependant of the Developer Package: it gets its data from there.
Corresponding classes* :	MSExcel MSProject
Relations :	Is a sub package of the main package Server. Is dependant of the Developer Package.
Sub packages :	Users

5.2.2.2 Level 2 Packages

Developer	
Description :	This package corresponds to the Developer Module. All information and methods regarding Developers are contained within this package.
Corresponding classes* :	Developers Timestamp Developer Identification
Relations :	Is a sub package of Users.
Sub packages :	None.

Administrator	
Description :	This package corresponds to the Administrator (Manager) Module. All information and methods regarding Administrators are contained within this package
Corresponding classes* :	Administrators Administrator Identification
Relations :	Is a sub package of Users.
Sub packages :	None.

5.2.3 Design Model: Design Class Diagrams



5.2.4 Design Classes Description

Property	Description
Name	Observable
Description	Abstract class representing the subject to be observed (Template Observer).
Responsibilities	None that need detailed information.
Relations	Abstract class, is a TMTServer Class Generalization.
Methods	AddObserver() : Add a new observer into subject Observers' list. DeleteObserver() : Delete an observer from the subject Observers' list. SetChanged() : Enable subject change. AlertObserver() : Alert engaged observers from subject change.
Attributes	None
Special Requirements	None

Property	Description
Name	Observer
Description	Abstract class representing the Subject's Observer (Template Observer).
Responsibilities	None that need detailed information.
Relations	Abstract class, is a DeveloperObserver & AdministratorObserver Generalization.
Methods	Update() : Indicates to the observer that the Subject has changed and to update his observations.
Attributes	None
Special Requirements	None

Property	Description
Name	DeveloperObserver
Description	Observer (Developer) implementation.
Responsibilities	Observes the TMTServer and alert client when the TMTServer changes.
Relations	Association with TMTServer & is an Observer Implementation.
Methods	Update() : Indicates to the DCM that the TMTServer has changed. Redraw() : Indicates to the DCM that its window must be refreshed.
Attributes	None
Special Requirements	None

Property	Description
Name	AdministratorObserver
Description	Observer (Administrator) implementation.
Responsibilities	Observes the TMTServer and alert client when the TMTServer changes.
Relations	Association with TMTServer & is an Observer Implementation.
Methods	Update() : Indicates to the MCM that the TMTServer has changed. Redraw() : Indicates to the MCM that its window must be refreshed.
Attributes	None
Special Requirements	None

Property	Description
Name	TMTServer
Description	Singleton and heart of the system. This class controls all other classes.
Responsibilities	Ensures that all queries from the clients are received and processed.
Relations	Is an Observable Implementation. Can be observed by Administrators and Developers. Association with Export and Import modules. Aggregation with Developers and Administrators.
Methods	ExecuteQuery() : Execute the query ReceiveQuery() : Listens to coming queries ValidateQuery() : Validates the query GenerateHTML() : Generates the query results
Attributes	AllDevs : Object: Developers' list. AllAdmins : Object: Administrators' list.
Special Requirements	None

Property	Description
Name	MSProject
Description	Transforms data coming from MSProject or data destined to be exported to MSProject.
Responsibilities	Ensures data importation from MSProject. Ensures data exportation to MSProject
Relations	Association with TMTServer. Dependant of Developers.
Methods	ReadFichier() : Read an MSProject file and add data to the database. WriteFile() : Write an MSProject file with data from the database.
Attributes	None
Special Requirements	None

Property	Description
Name	MSExcel
Description	Transforms data coming from MSExcel or data destined to be exported to MSExcel.
Responsibilities	Ensures data importation from MSExcel. Ensures data exportation to MSExcel
Relations	Association with TMTServer. Dependant of Developers.
Methods	ReadFichier() : Read an MSExcel file and add data to the database. WriteFile() : Write an MSExcel file with data from the database.
Attributes	None
Special Requirements	None

Property	Description
Name	Administrators
Description	Singleton containing Administrators' list.
Responsibilities	Maintain an updated list of all Administrators.
Relations	Aggregation to TMTServer. Aggregation to Administrators.
Methods	AddAdministrator() : Add an Administrator to the system DeleteAdministrator() : Delete an Administrator from the system. ModifyAdministrator() : Modify an Administrator in the system. ValidateLogin() : Validates username and password.
Attributes	AllAdministrators : List of all system's Administrators.
Special Requirements	None

Property	Description
Name	Developers
Description	Singleton containing Developers' list.
Responsibilities	Maintain an updated list of all Developers.
Relations	Aggregation to TMTServer. Aggregation to Developers.
Methods	AddDeveloper() : Add an Developer to the system DeleteDeveloper () : Delete a Developer from the system. ModifyDeveloper () : Modify a Developer in the system. ValidateLogin() : Validates username and password.
Attributes	AllAdministrators : List of all system's Developers.
Special Requirements	None

Property	Description
Name	Identification
Description	Parent class representing a system user (Administrator or Developer).
Responsibilities	None
Relations	Is an Administrator or a Developer Generalization.
Methods	GetNoID() : Returns user's ID number. SetNoID() : Modifies user's ID number. GetName() : Returns username. SetName() : Modifies username. GetPW() : Returns user's password. SetPW(): Modify user's password.
Attributes	NoID : User's ID number. Name : Username. PW : User's Password.
Special Requirements	None

Property	Description
Name	Administrator
Description	Represents the Administrator entity.
Responsibilities	Models the Administrator entity.
Relations	Aggregation to Administrators and derived from Identification. An Administrator is in the Administrators' list and is based upon a user's Identification.

Methods	None
Attributes	None – Inherited attributes from Identification.
Special Requirements	None

Property	Description
Name	Developer
Description	Represents the Developer entity.
Responsibilities	Models the Developer entity.
Relations	Aggregation to Developers and derived from Identification. A Developer is in the Developers' list and is based upon a user's Identification.
Methods	AddTimestamp() : Add a timestamp. DeleteTimestamp() : Delete a timestamp. ModifyTimestamp() : Modify a timestamp. GetWeekTimestamps() : Return current week's timestamps. GetAllTimestamps() : Return all timestamps. GetNoAdmin() : Return Developer's Administrator ID number. SetNoAdmin() : Modify Developer's Administrator ID number. GetProjects() : Return Projects' list. SetProjects() : Modify Projects' list. GetTasks() : Return Tasks' list. SetTasks() : Modify Tasks' list. GetActivities() : Return Activities' list. SetActivities() : Modify Activities' list. GetArtifacts() : Return Artifacts' list. SetArtifacts(): Modify Artifacts' list.
Attributes	allTimestamps : Timestamps' list. NoAdmin : Developer's Administrator ID number. Projects : Projects' list. Tasks : Tasks' list. Activities : Activities' list. Artifacts : Artifacts' list.
Special Requirements	None

Property	Description
Name	Timestamp
Description	Represents the Timestamp entity.
Responsibilities	Models the Timestamp entity.
Relations	Aggregation to Developer. A Developer has 0 or more timestamps.
Methods	GetNoID() : Return Timestamp's ID SetNoID() : Modify Timestamp's ID. GetNoDev() : Return Timestamp's associated Developer ID. SetNoDev() : Modify Timestamp's associated Developer ID. GetProject() : Return Timestamp's associated Project. SetProject() : Modify Timestamp's associated Project. GetWorker() : Return Timestamp's associated Role (worker). SetWorker() : Modify Timestamp's associated Role (worker).. GetActivity() : Return Timestamp's associated Activity. SetActivity() : Modify Timestamp's associated Activity. GetArtifact() : Return Timestamp's associated Artifact. SetArtifact() : Modify Timestamp's associated Artifact. GetDate() : Return Timestamp's Date. SetDate() : Modify Timestamp's Date. GetStartTime() : Return Timestamp's Starting Hour. SetStartTime() : Modify Timestamp's Starting Hour. GetEndTime() : Return Timestamp's Ending Hour. SetEndTime() : Modify Timestamp's Ending Hour. GetParticipants() : Return Timestamp's Participants. SetParticipants() : Modify Timestamp's Participants. GetDescription() : Return Timestamp's Description. SetDescription() : Modify Timestamp's Description.
Attributes	NoID : Timestamp's ID. NoDev : Timestamp's associated Developer ID Project : Timestamp's associated Project. Worker : Timestamp's associated Role (worker).. Activity : Timestamp's associated Activity. Artifact : Timestamp's associated Artifact. Date : Timestamp's Date. StartTime : Timestamp's Starting Hour. EndTime : Timestamp's Ending Hour. Participants : Timestamp's Participants. Description : Timestamp's Description.
Special Requirements	None

6. Interface Description

See User Interface documents - upedu_ex_uiprt.pdf

7. Size and Performance

The selected architecture supports the sizing and timing requirements through the implementation of a client-server architecture. The client portion is implemented on local campus PCs or remote dial up PCs. The components have been designed to ensure that minimal disk and memory requirements are needed on the PC client portion.

8. Quality

The software architecture supports the quality requirements, as stipulated in the Software Requirements Specification and Supplementary Specification

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitor Tool
Use-Case-Realization Specification

Version <5.0>

Revision History

Date	Version	Description	Author
15/01/2001	1.0	First Version	Sandra Lee
15/01/2001	2.0	Added Objects Diagrams	Sandra Lee
06/02/2001	3.0	Modified content to fit the new Use-Case Diagrams	Sandra Lee
30/03/2001	4.0	Adding Validate Timestamps Functionality	James Prevost
05/04/2001	5.0	Adding Classes: ShowWindow and CloseWindow. Revised Objects Diagrams.	Sandra Lee

Preface

The following case study has been modified from its original content. The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Analysis & Design Section.

Regards,

Unified Process for Education Team

Table of Contents

Figures

Use-Case-Realization Specification

1. Introduction

1.1 Purpose

This document provides a comprehensive overview of the system, using a number of different diagrams for representing the system functions.

1.2 Scope

The Time Monitoring Tool system allows developers working within a defined software development process to record the time spent on the various activities, in a database. The TMT will also allow a manager to derive analyses and produce reports based on the data entered in the system. This Use Case Realization document provides an overview of the use cases developed in Time Monitoring Tool.

1.3 Definitions, Acronyms, and Abbreviations

See Glossary, document upedu_ex_gloss.doc

1.4 References

1. TMT - Glossary
2. TMT - Use Case Specification
3. TMT - Supplementary Specification
4. TMT - Iteration Plan

1.5 Overview

The sections of the Use-Case Realization document describes use cases in terms of their flow of events, participant objects and corresponding diagrams.

1. USE CASE <Load MCM >

1.1 Brief Description

This Use-Case defines how the MCM is loaded allowing the Administrator to manage the application through the Internet Browser.

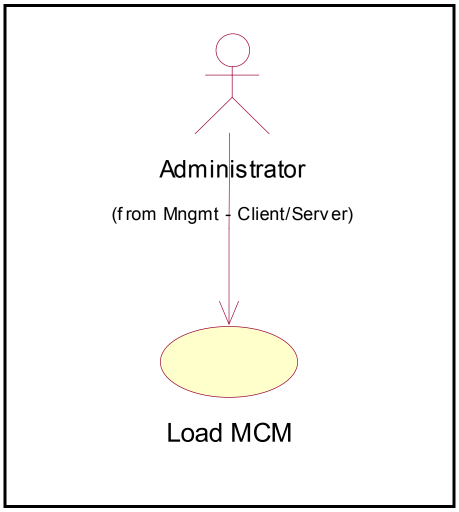


Figure 1 : Load MCM

1.2 Flow Events

Upon logging in the system, the Administrator implicitly makes a Load MCM query which is received by the Browser, then executed.

1.3 Interaction Diagrams

- The Administrator launches the Load MCM query.
- The Internet Browser receives the query and loads the module.

1.3.1. Sequence Diagrams

This Sequence Diagram shows Actors and Objects messages exchange in the Use-Case < Load MCM >.

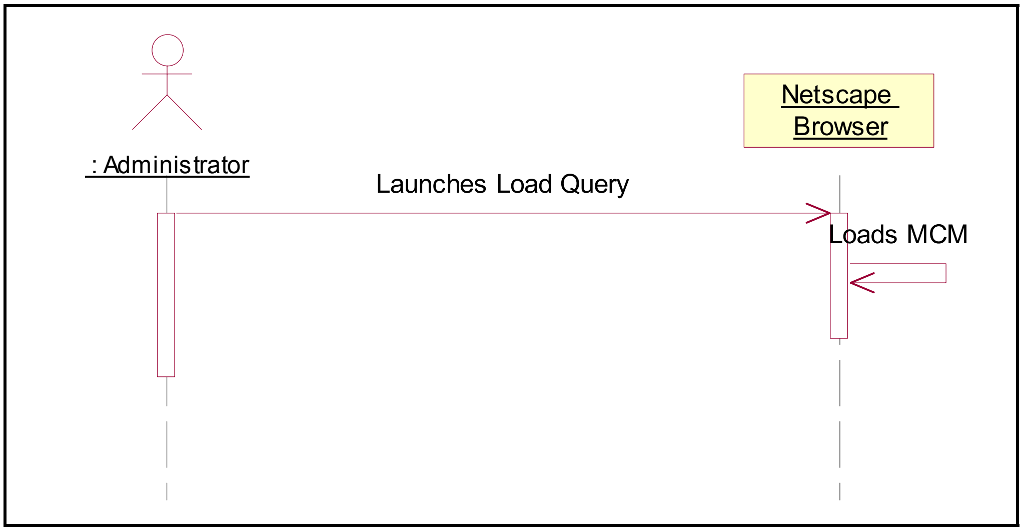


Figure 2 : Sequence Diagram : Load MCM

1.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case < Load MCM >.

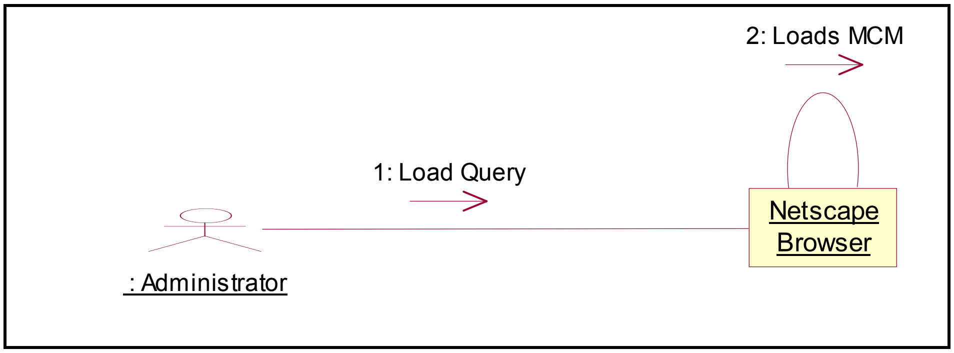


Figure 3 : Collaboration Diagram : Load MCM

1.6 Participating Objects

The following objects collaborate and define the Use-Case < Load MCM > Behavior:

Netscape Browser **This object represents the visible part of the application and allows the Administrator to load his module.**

1.7 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

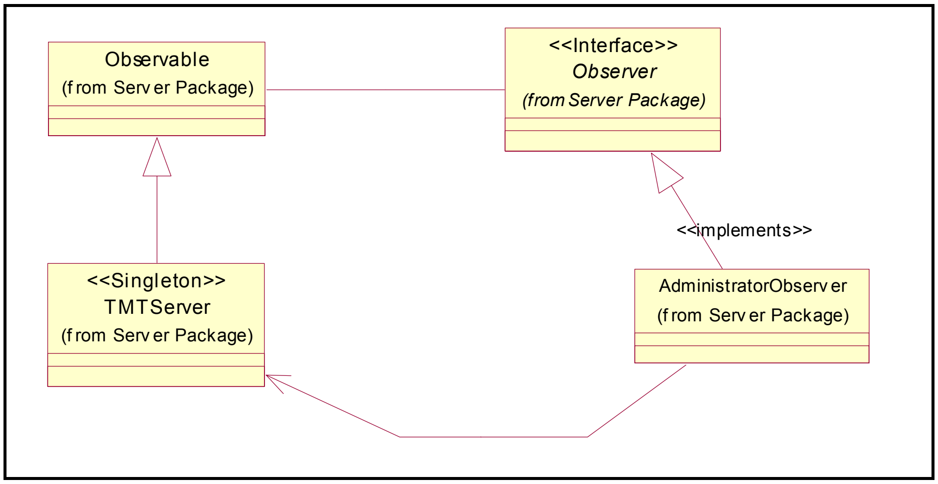


Figure 4 : Object Diagrams : Load MCM

2. USE CASE <Load DCM >

2.1 Brief Description

This Use-Case defines how the DCM is loaded allowing the Developer to timestamp his work hours through the Internet Browser.

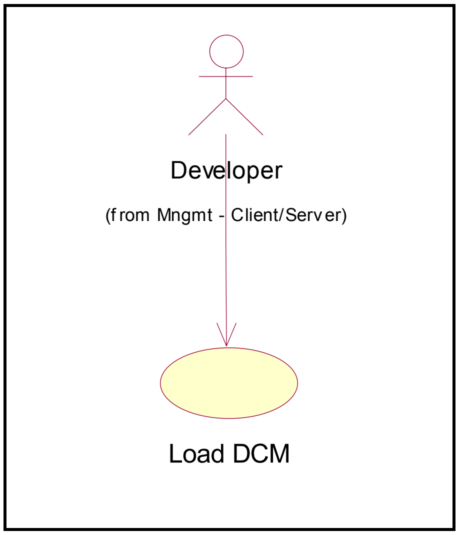


Figure 5 : Load DCM

2.2 Flow Events

Upon logging in the system, the Developer implicitly makes a Load DCM query which is received by the Browser, then executed.

2.3 Interaction Diagrams

- **The Developer launches the Load DCM query.**
- **The Internet Browser receives the query and loads the module.**

2.3.1 Sequence Diagrams

This Sequence Diagram shows Actors and Objects messages exchange in the Use-Case < Load DCM >.

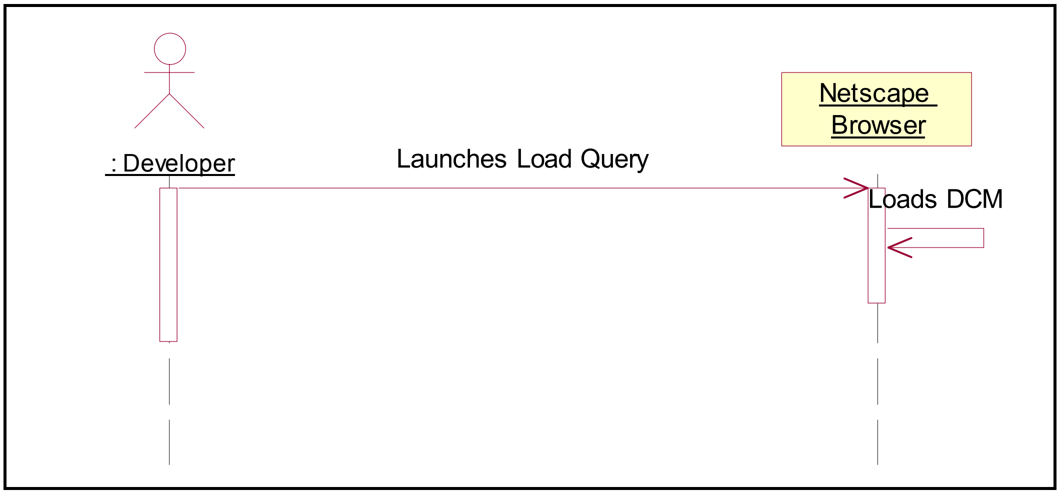


Figure 6 : Sequence Diagram : Load DCM

2.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case -<Load DCM>.

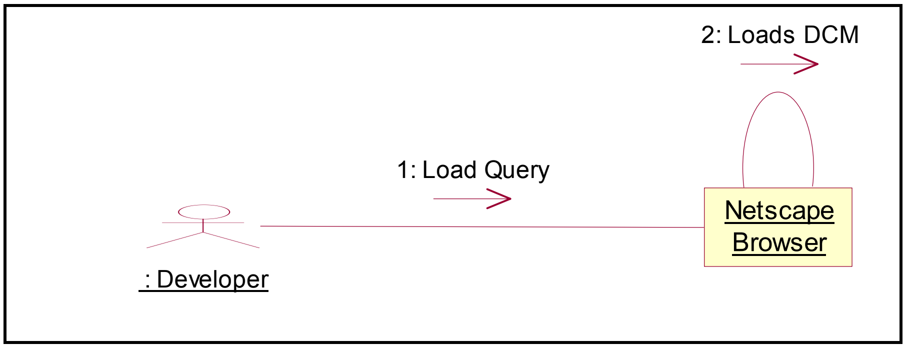


Figure 7 : Collaboration Diagram : Load DCM

2.4 Participating Objects

The following objects collaborate and define the Use-Case < Load DCM > Behavior:

<u>Netscape Browser</u>	**This object represents the visible part of the application and allows the Developer to load his module.**

2.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

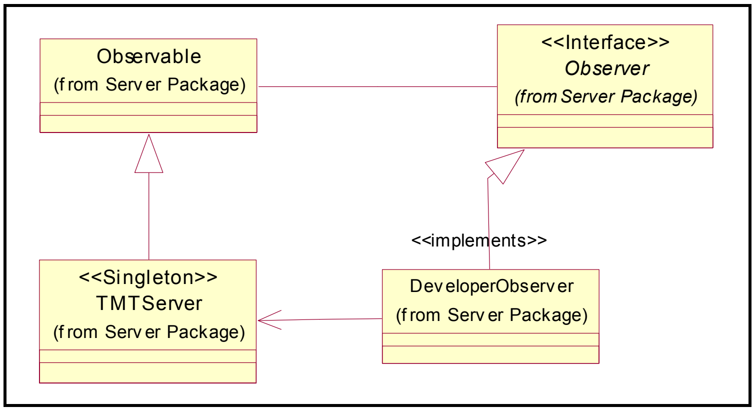


Figure 8 : Class Diagram : Load DCM

3. USE CASE <Login>

3.1 Brief Description

This Use-Case defines how users are logged into the system and get access to their respective functionalities.

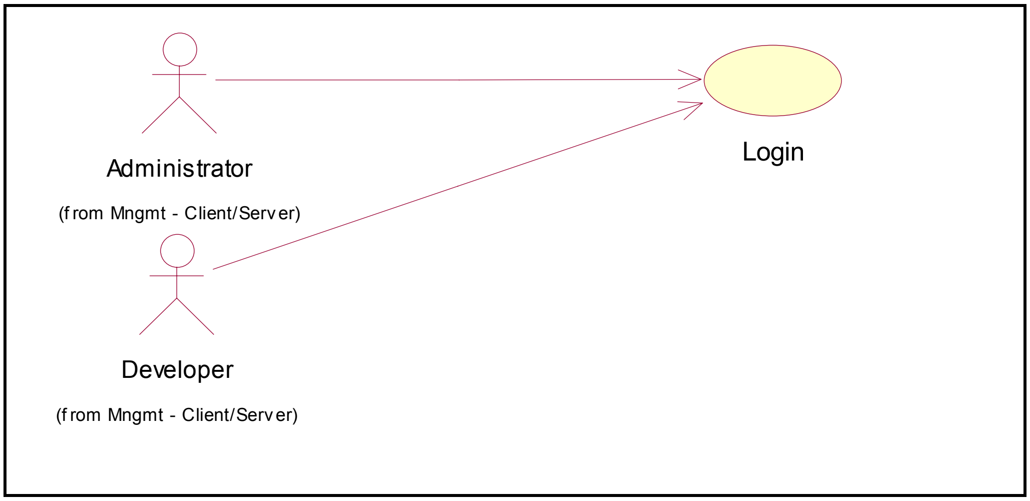


Figure 8: Login

3.2 Flow Events

The user provides his username and password and submits the form. Data is validated and login process is activated.

3.3 Interaction Diagrams

- **The user enters his username and password and submits the data**
- **Query is received by the MCM or DCM and transmitted to the SM**
- **The SM validates, executes and accesses the database for login confirmation.**
- **The DBMS returns confirmation to the SM which transmits it to the MCM.**

3.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case < Login>.

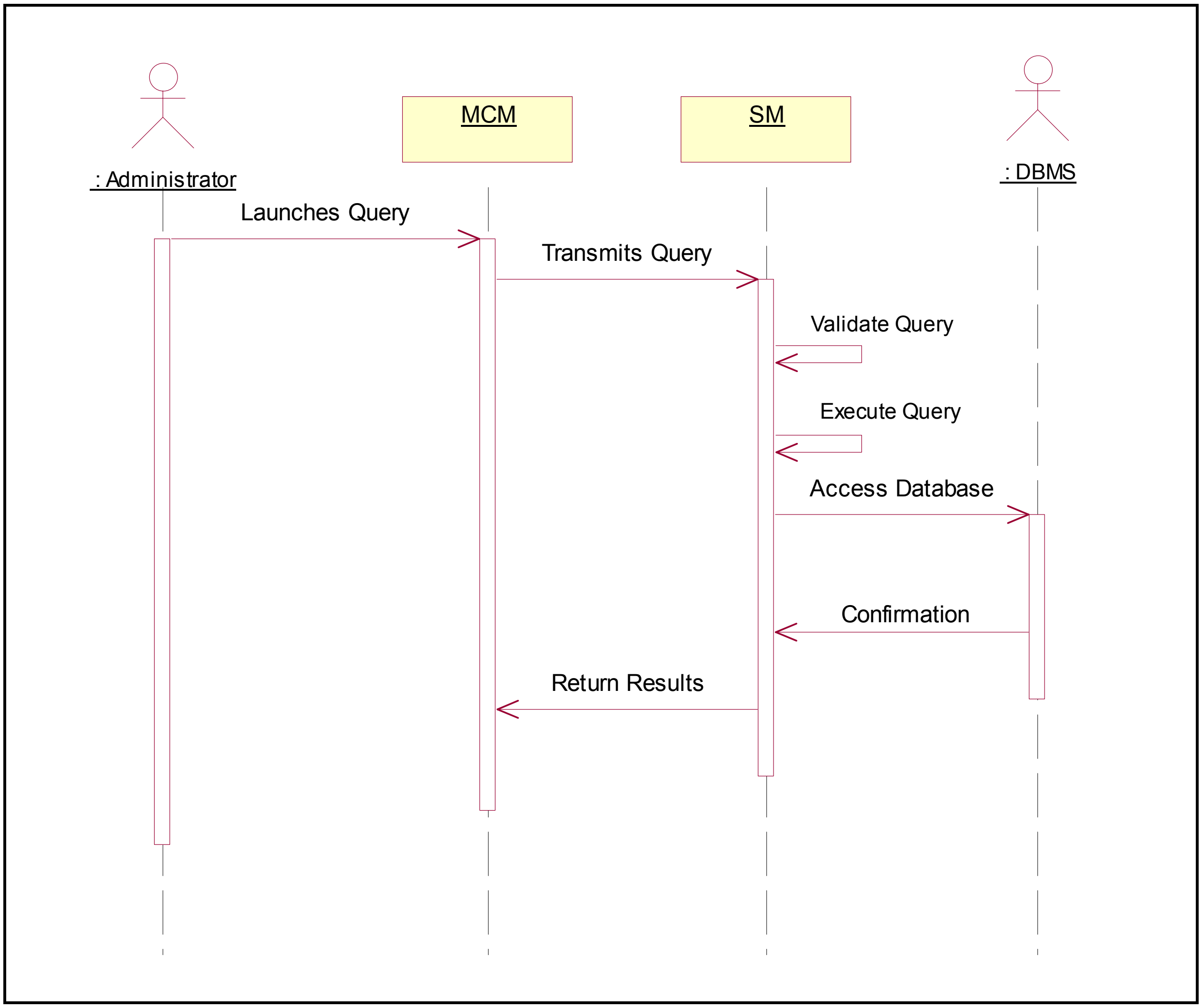


Figure 9 : Sequence Diagram : Login(Administrator)

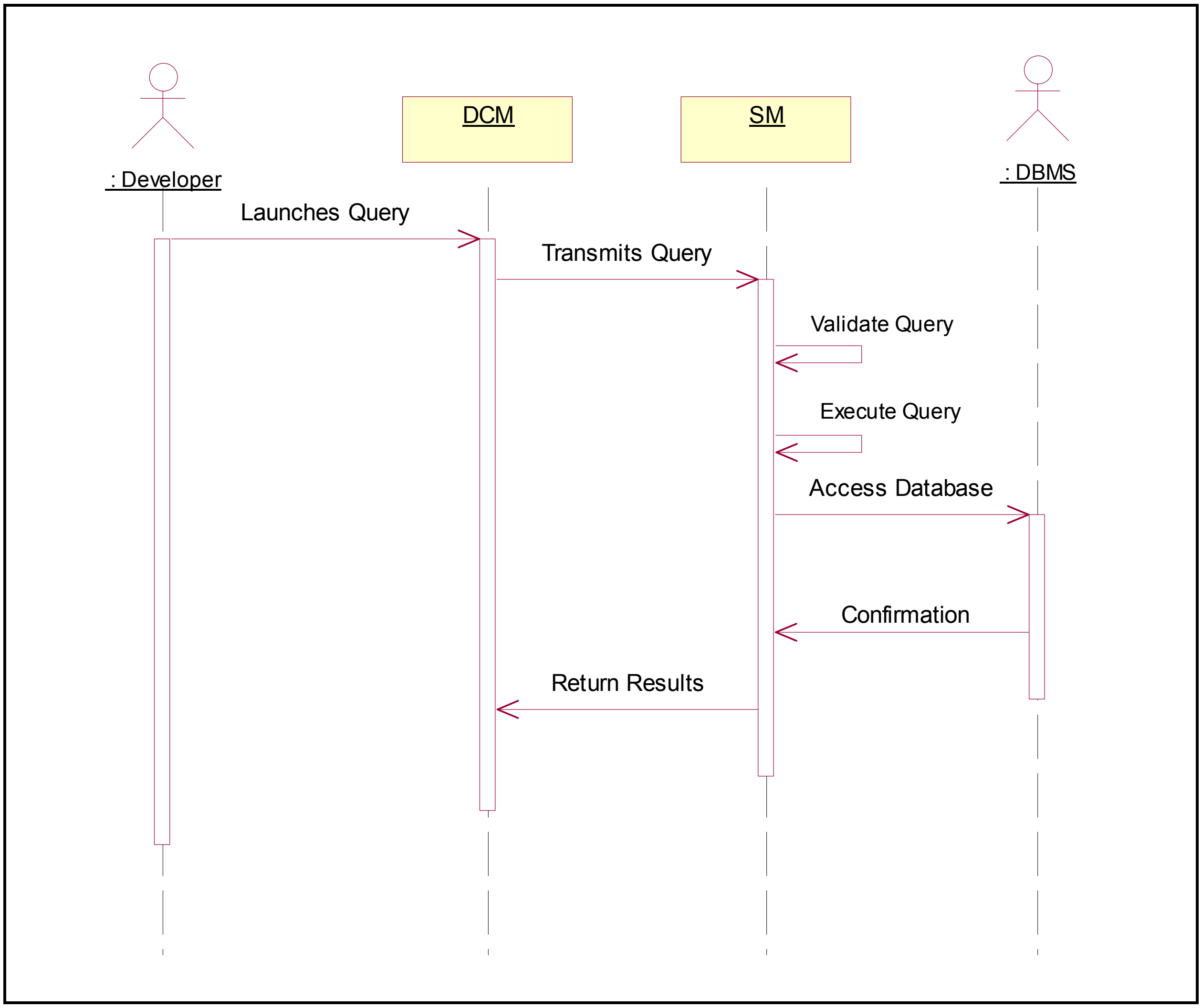


Figure 10 : Sequence Diagram : Login(Developer)

3.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <login>.

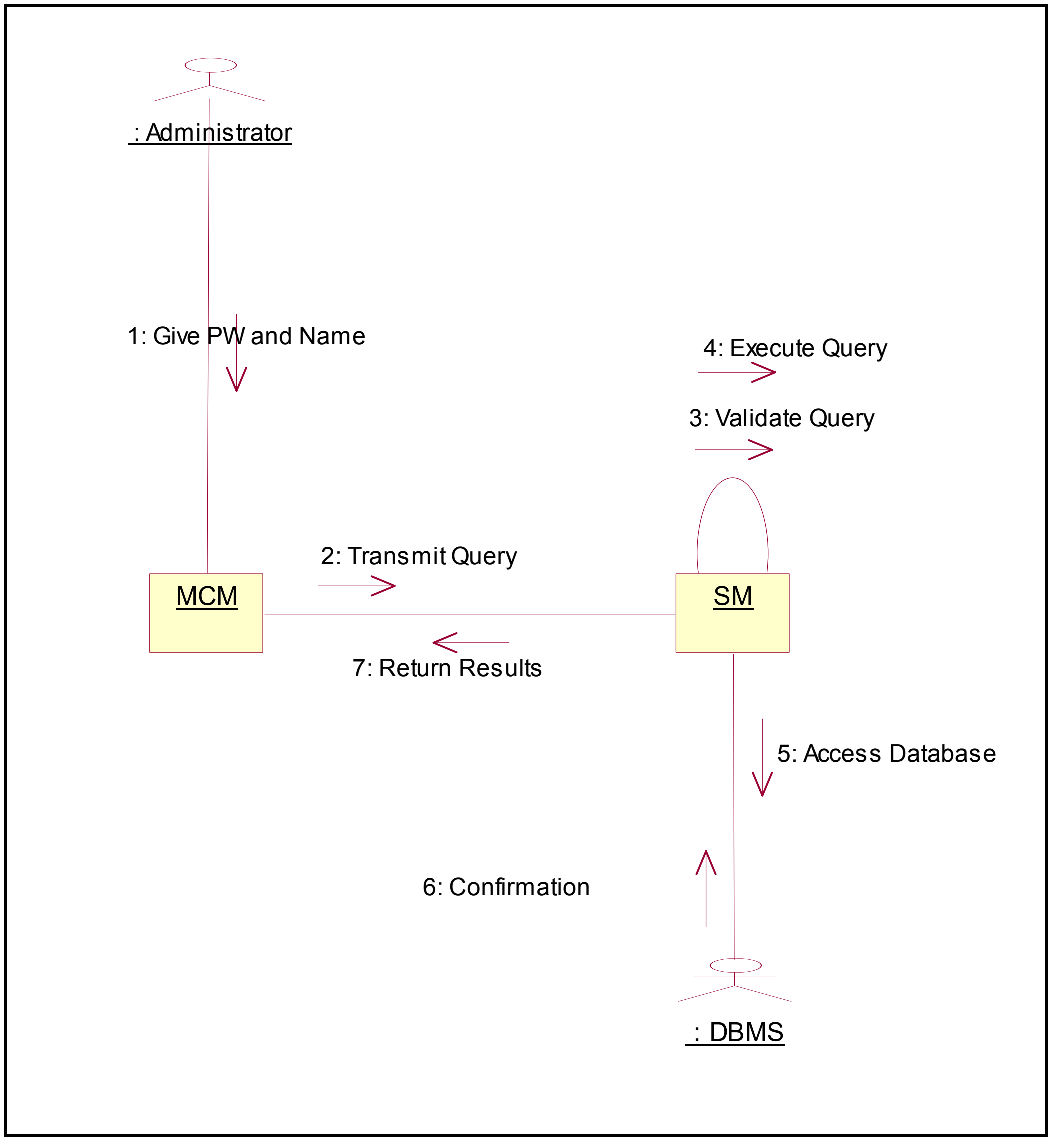


Figure 11 : Collaboration Diagram : Login (Administrator)

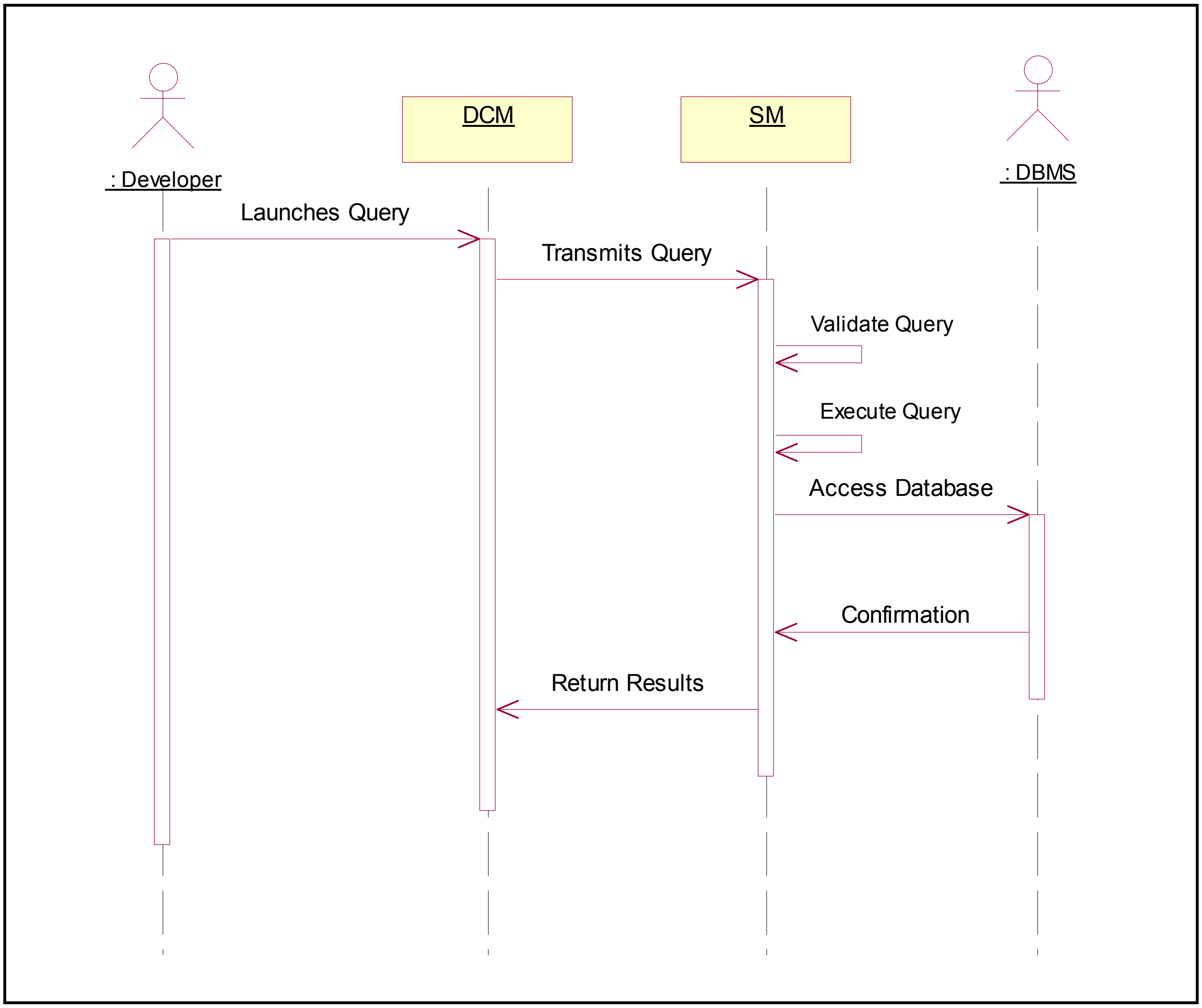


Figure 12 : Collaboration Diagram : Login (Developer)

3.4 Participating Objects

The following objects collaborate and define the Use-Case <login> :

MCM This object represents the visible part of the application and allows the Administrator to login to the system.

DCM This object represents the visible part of the application and allows the Developer to login to the system.

SM This object executes and validates the Login query by communicating with the Database.

3.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

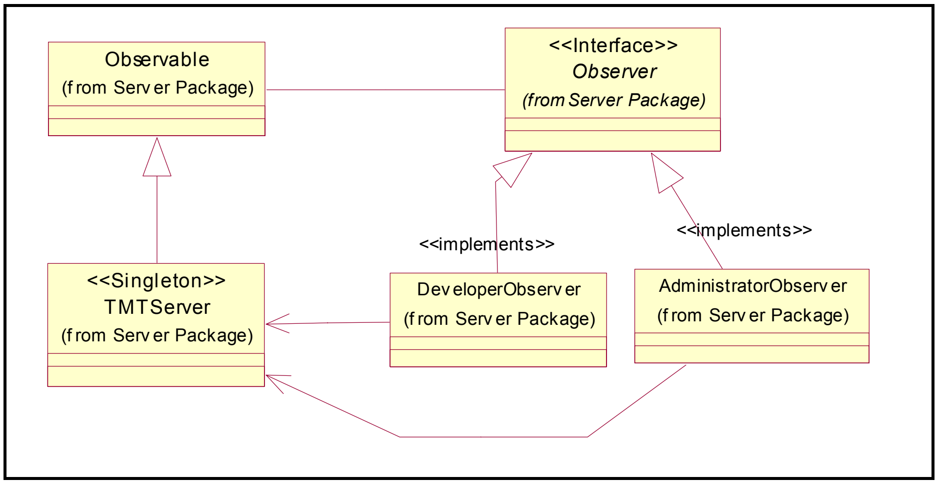


Figure 13 : Object Diagram : Login

4. USE CASE <Logout>

4.1 Brief Description

This Use-Case defines how users are logged out from system and exit the application normally.

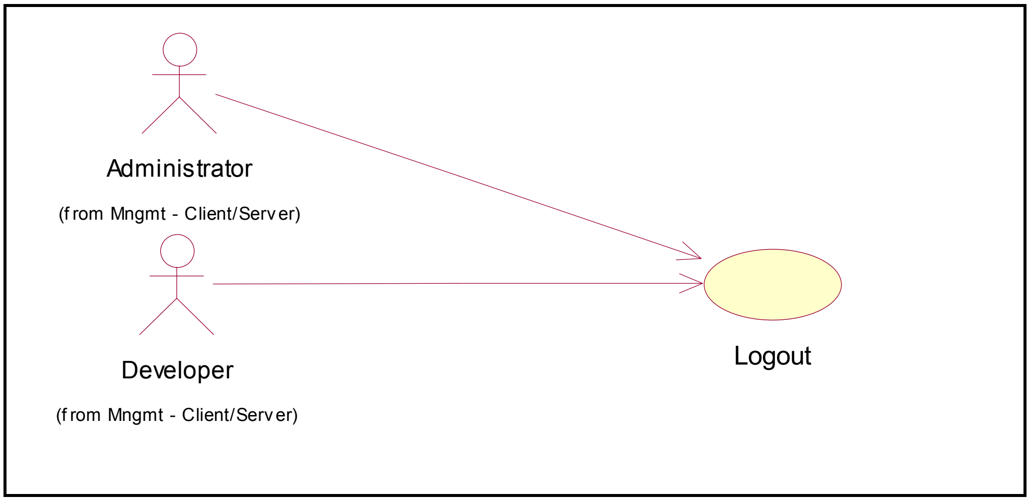


Figure 14 : Logout

4.2 Flow Events

The user exits the application by using the appropriate End Session button. Query is validated and logout process is activated.

4.3 Interaction Diagrams

- **The user clicks the End Session Button**
- **Query is received by the MCM or DCM and transmitted to the SM**
- **The SM validates, executes and accesses the database for logout confirmation.**
- **The DBMS returns confirmation to the SM which transmits it to the MCM.**

4.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case < Logout>.

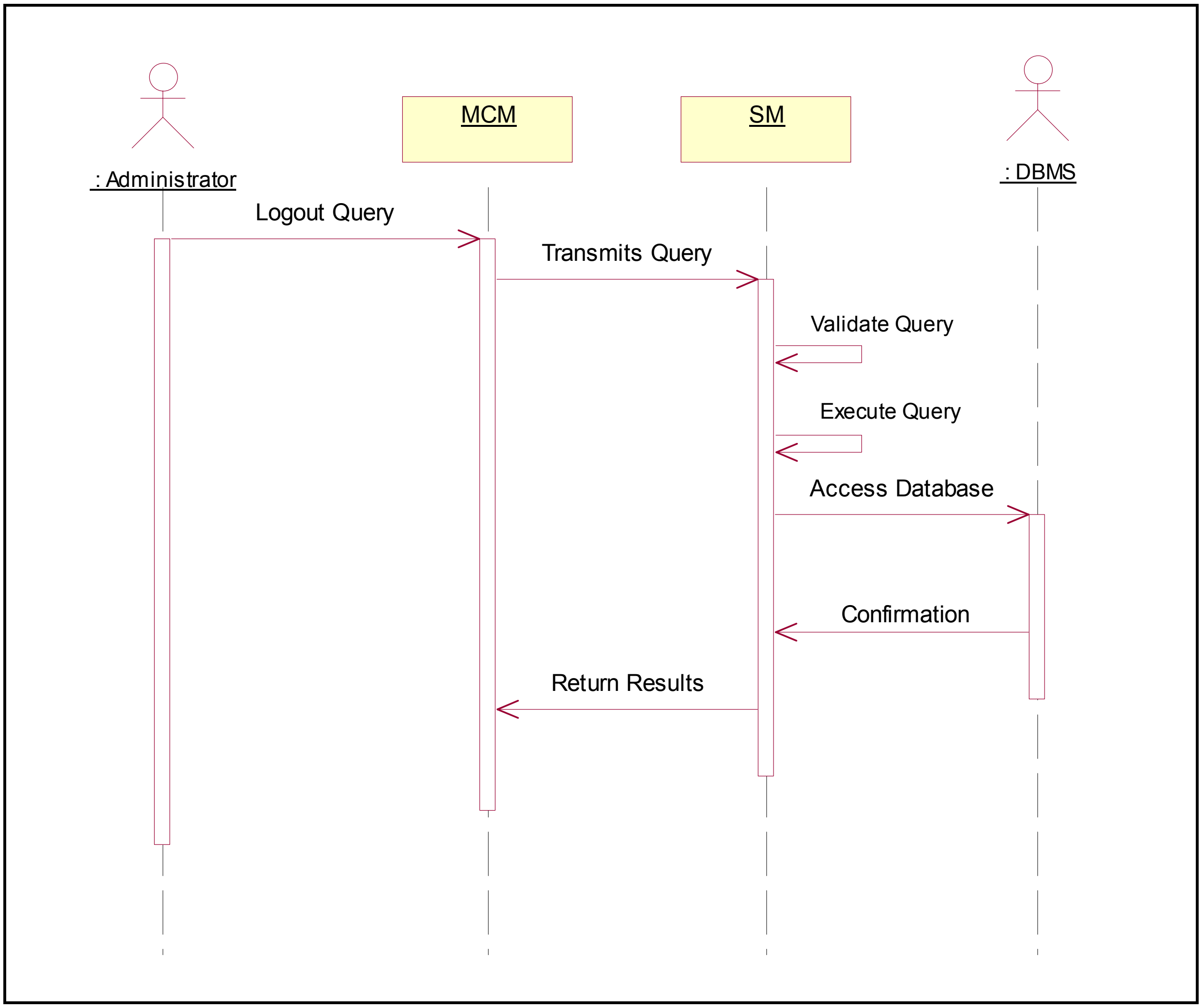


Figure 15 : Sequence Diagram : Logout (Administrator)

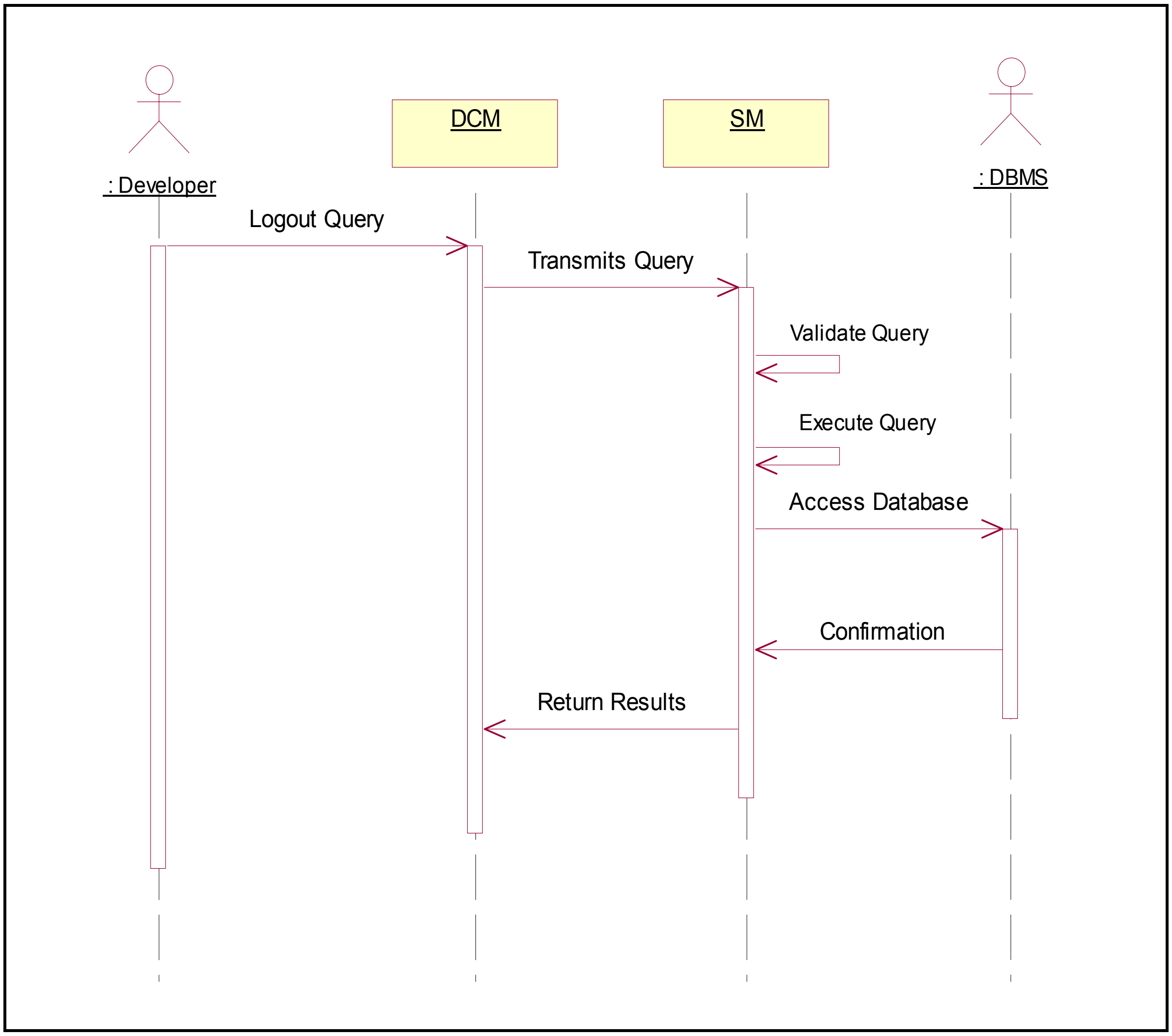


Figure 16 : Sequence Diagram : Logout (Developer)

4.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <Logout>

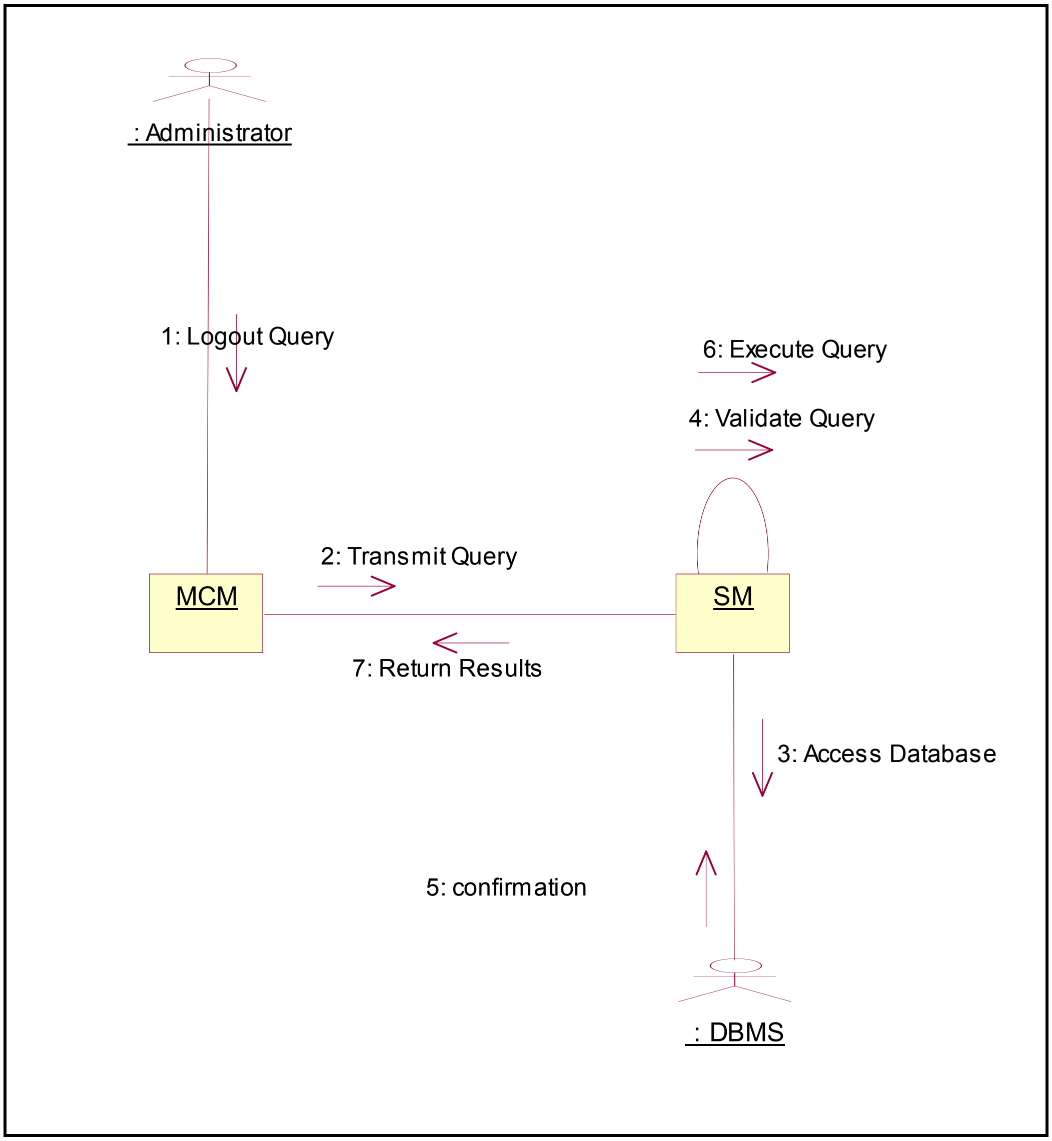


Figure 17 : Collaboration Diagram : Logout (Administrator)

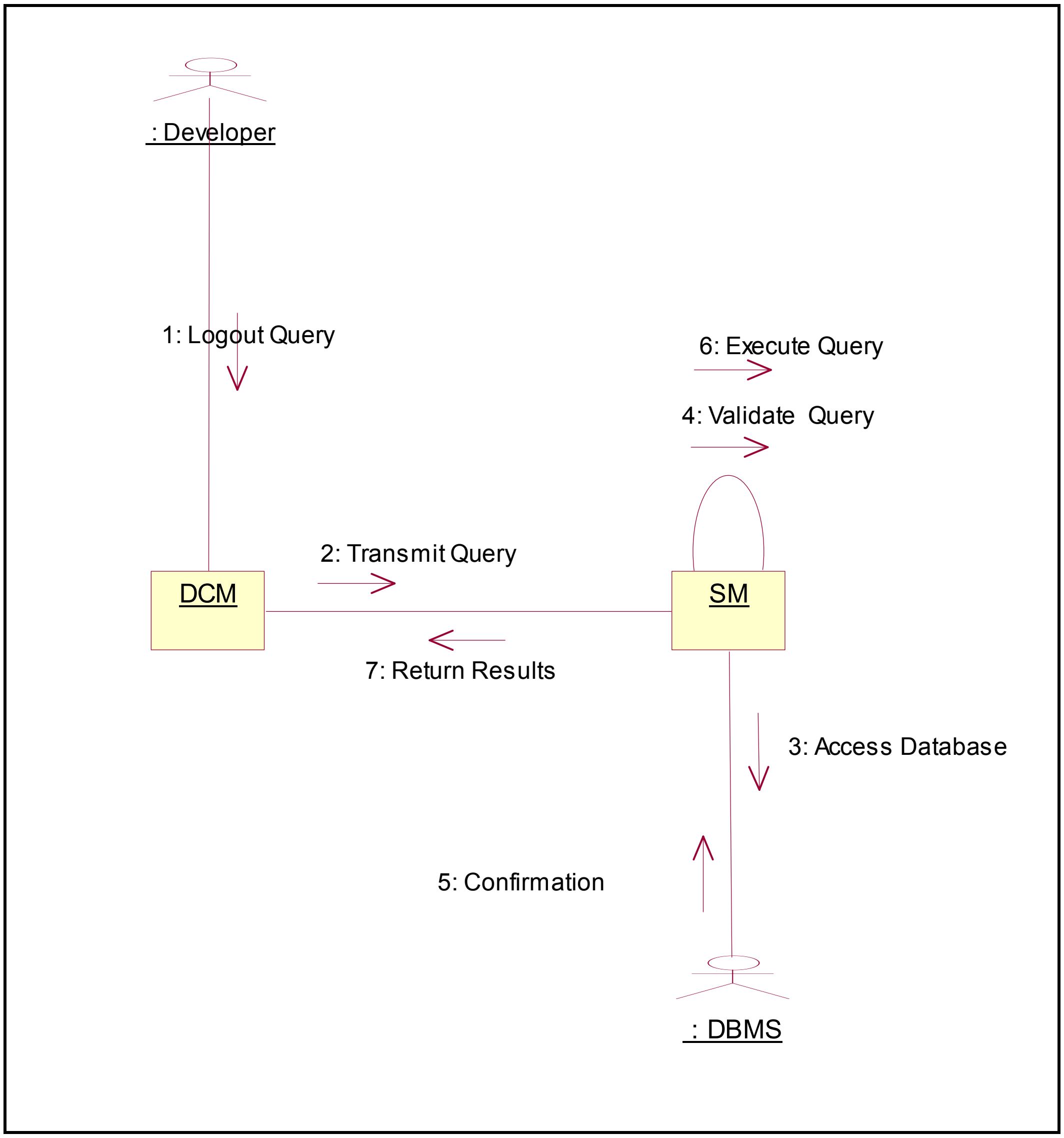


Figure 18 : Collaboration Diagram : Logout (Developer)

4.4 Participating Objects

The following objects collaborate and define the Use-Case <Logout> :

MCM	**This object represents the visible part of the application and allows the Administrator to logout from the system.**
DCM	**This object represents the visible part of the application and allows the Developer to logout from the system.**
SM	**This object executes and validates the Logout query by communicating with the Database.**

4.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

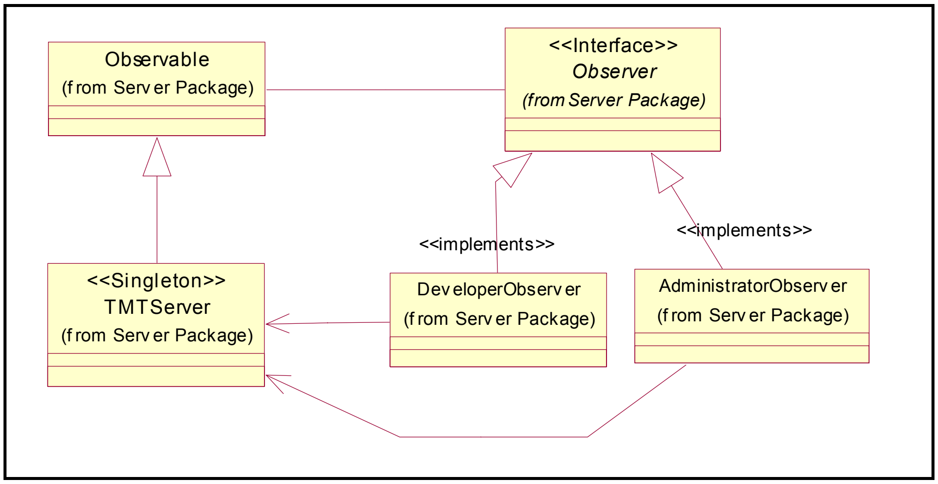


Figure 19 : Object Diagrams : Logout

5. USE CASE <Show TMT Status >

5.1 Brief Description

This Use-Case defines how the TMT Status is displayed to the user.

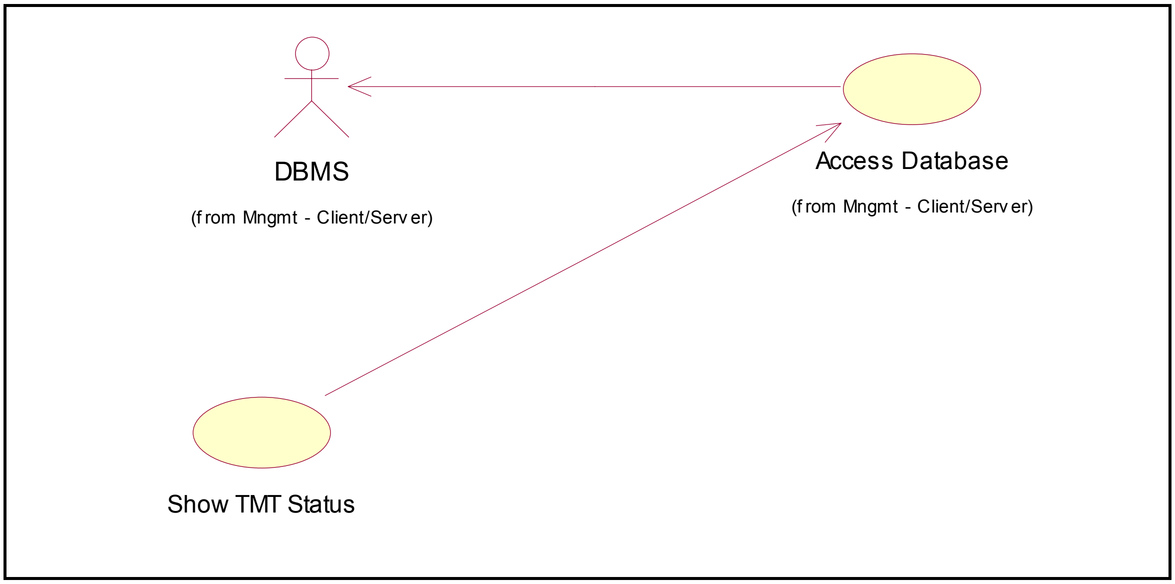


Figure 20 : Show TMT Status

5.2 Flow Events

The TMT status is constantly verified by the Server Module. If the status is abnormal, error details and comments will be displayed, else the normal TMT Window content is displayed.

5.3 Interaction Diagrams

- **The SM verifies the status**
- **The SM shows the status**

5.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case < Show TMT Status>.

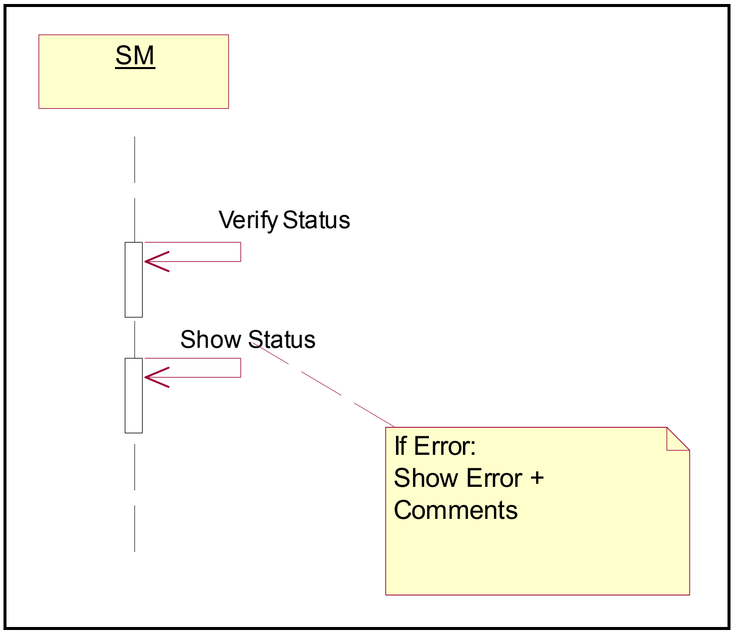


Figure 21 : Sequence Diagram : Show TMT Status

5.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <Show TMT Status>.

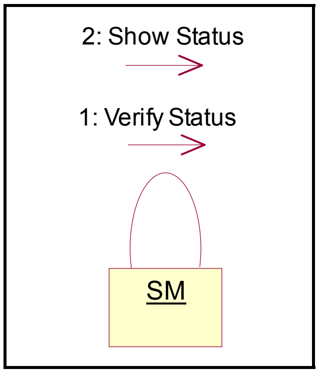


Figure 22 : Collaboration Diagram : Show TMT Status

5.4 Participating Objects

The following objects collaborate and define the Use-Case < Show TMT Status > :

SM **This object interacts with the database and the Internet Browser in order to determine the current status of the TMT.**

5.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

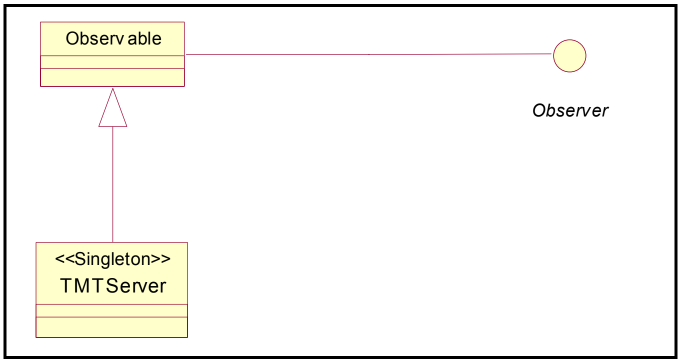


Figure 23 : Object Diagrams : Show TMT Status

6. USE CASE <Identify Errors >

6.1 Brief Description

This Use-Case defines the TMT status error identification process

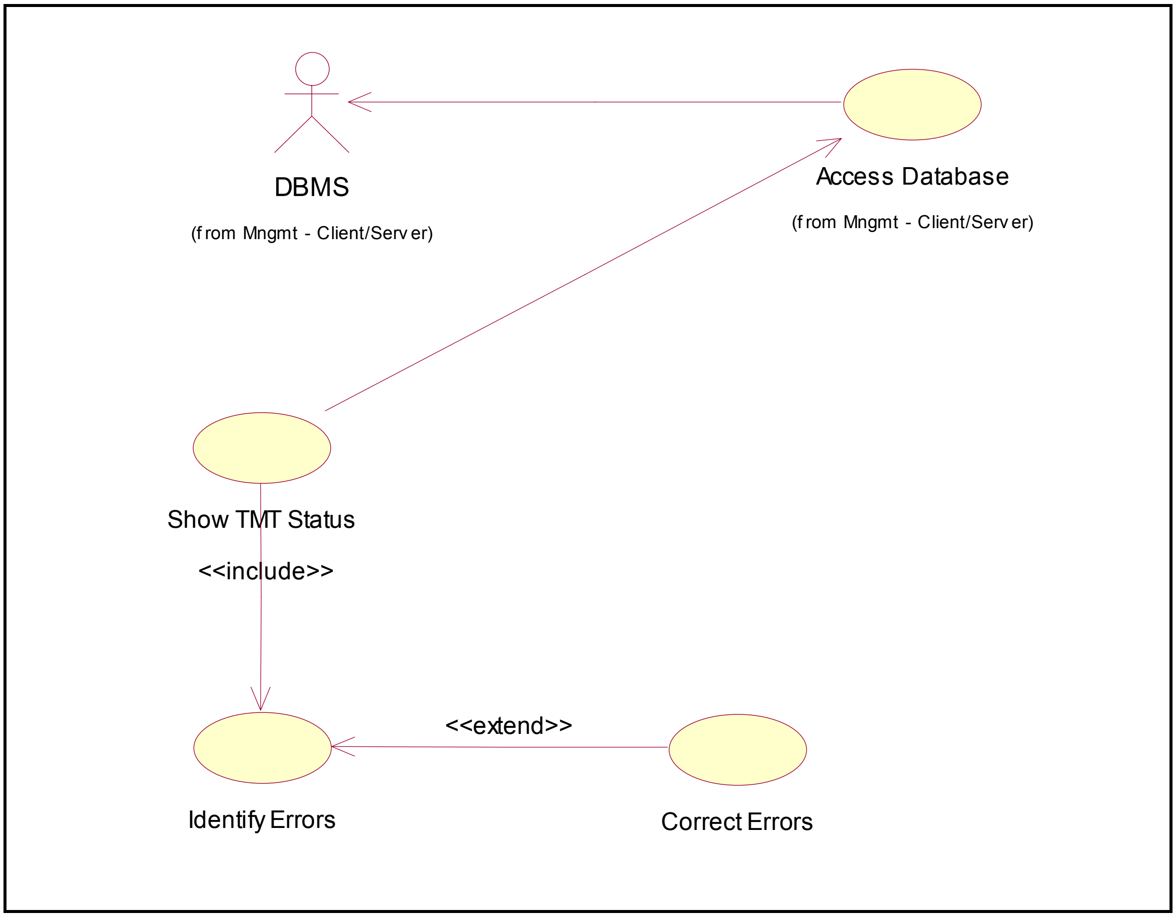


Figure 24 : Identify Errors

6.2 Flow Events

Upon each TMT status verification, the SM must detect any error presence (Execution, Communication, Validation, others).

6.3 Interaction Diagrams

- **The SM verifies the TMT status**
- **The SM identifies errors, if any**

6.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case < Identify Errors>.

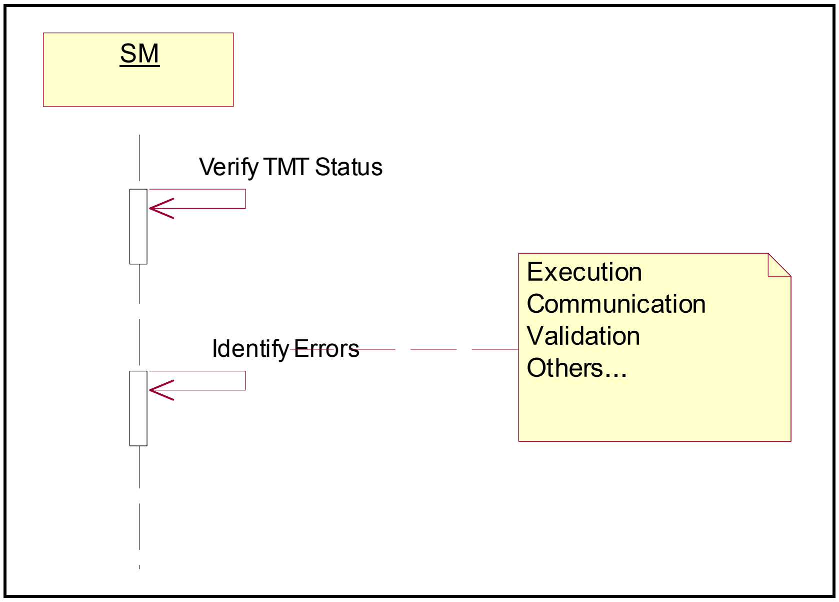


Figure 25 : Sequence Diagram : Identify Errors

6.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <Identify Errors>.

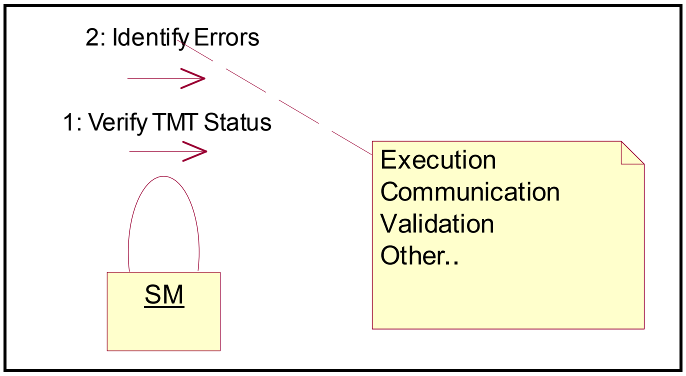


Figure 26 : Collaboration Diagram : Identify Errors

6.4 Participating Objects

The following objects collaborate and define the Use-Case < Identify Errors> :

<u>SM</u> **This object interacts with the database and the Internet Browser in order to determine and detect error presence.**

6.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

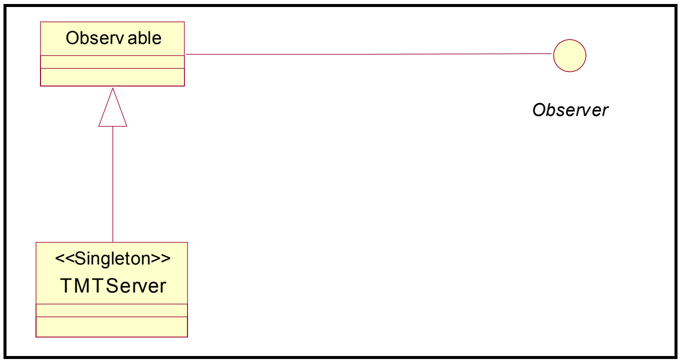


Figure 27 : Object Diagrams : Identify Errors

7. USE CASE < Correct Errors >

7.1 Brief Description

After Identifying errors, the SM must try to automatically correct the error, if possible. The error correction must remain transparent to the user and the application must follow its course of actions normally.

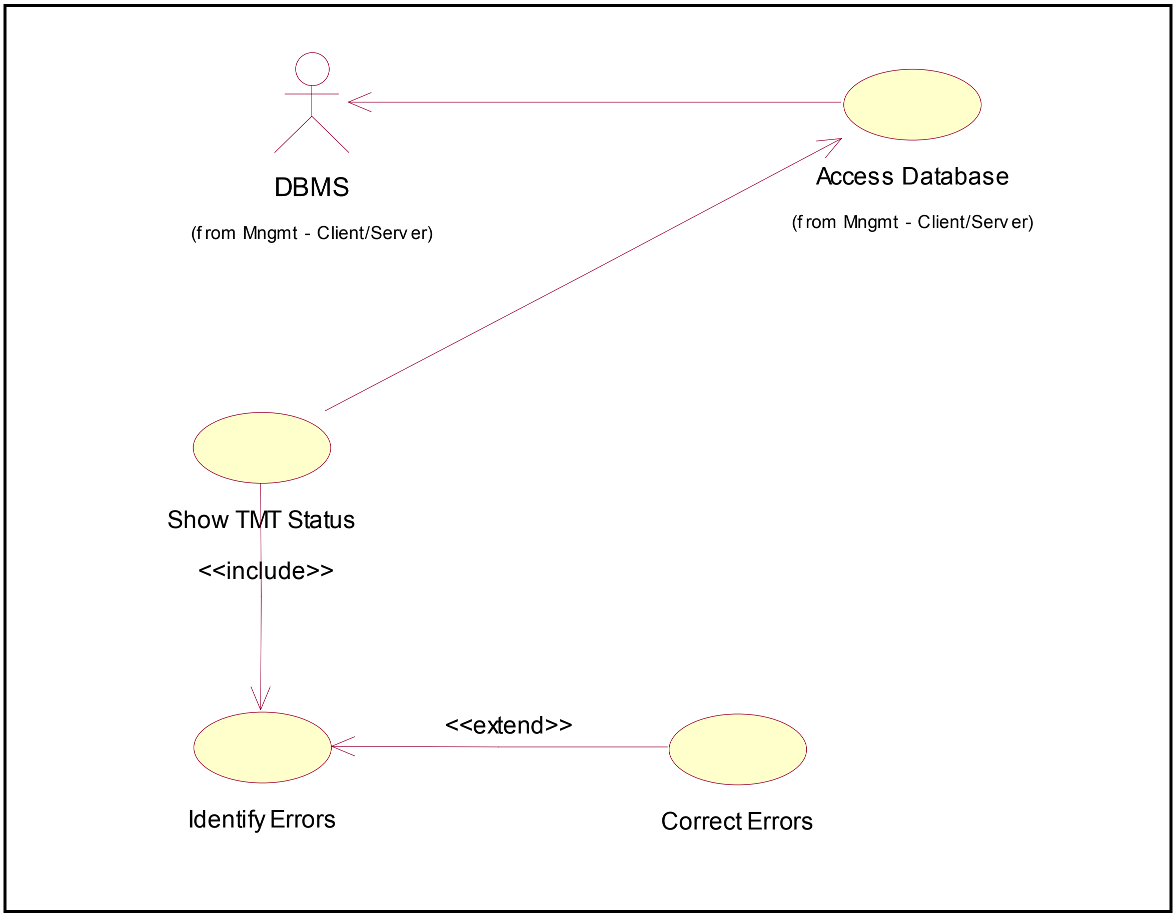


Figure 28 : Correct Errors

7.2 Flow Events

Upon each TMT status error identification, the SM must try to correct the error.

7.3 Interaction Diagrams

- **The SM gets the error information from its identification**
- **The SM tries to correct the error**
- **If the error is corrected, the SM restores the TMT last good status**
- **If the SM cannot correct the error, a special TMT status error message is displayed with appropriate comments**

7.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case < Correct Errors>.

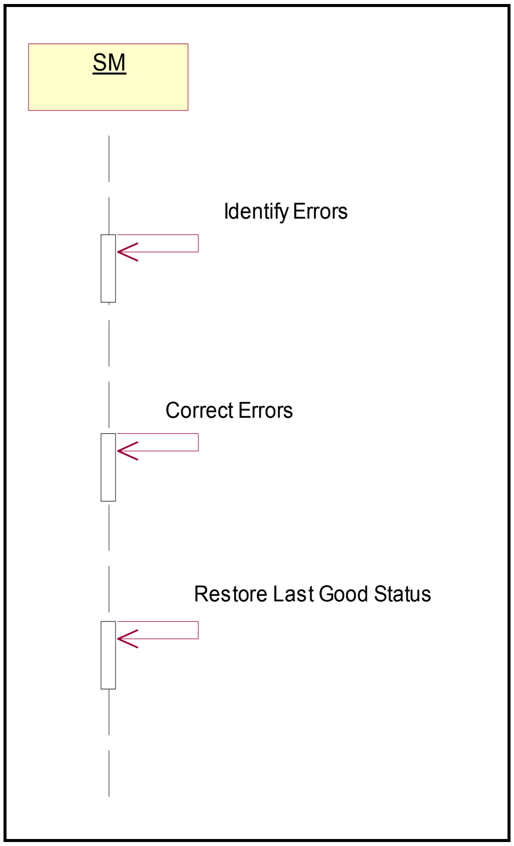


Figure 29 : Sequence Diagram : Correct Errors

7.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <Correct Errors>.

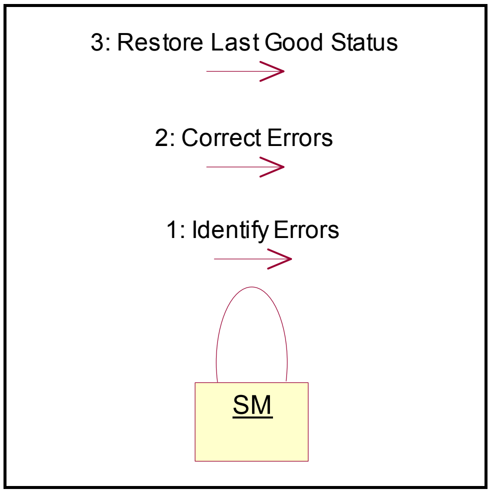


Figure 30 : Collaboration Diagram : Correct Errors

7.4 Participating Objects

The following objects collaborate and define the Use-Case < Correct Errors> :

SM **This object interacts with the database and the Internet Browser in order to correct the errors.**

7.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

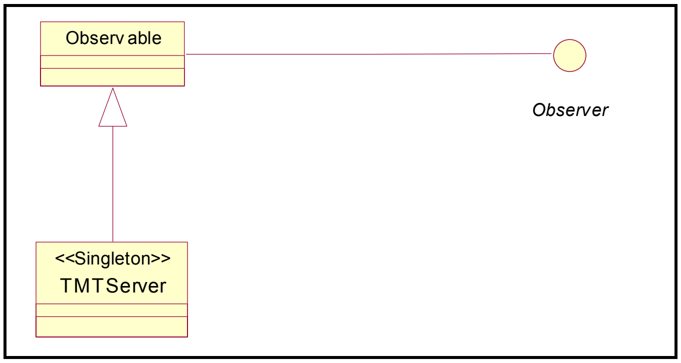


Figure 31 : Object Diagrams : Correct Errors

8. USE CASE < Show TMT Window >

8.1 Brief Description

This Use-Case defines how the appropriate TMT Window (Manager Client Window or Developer Client Window) is displayed on screen, using the data provided on the Login Screen (username and password).

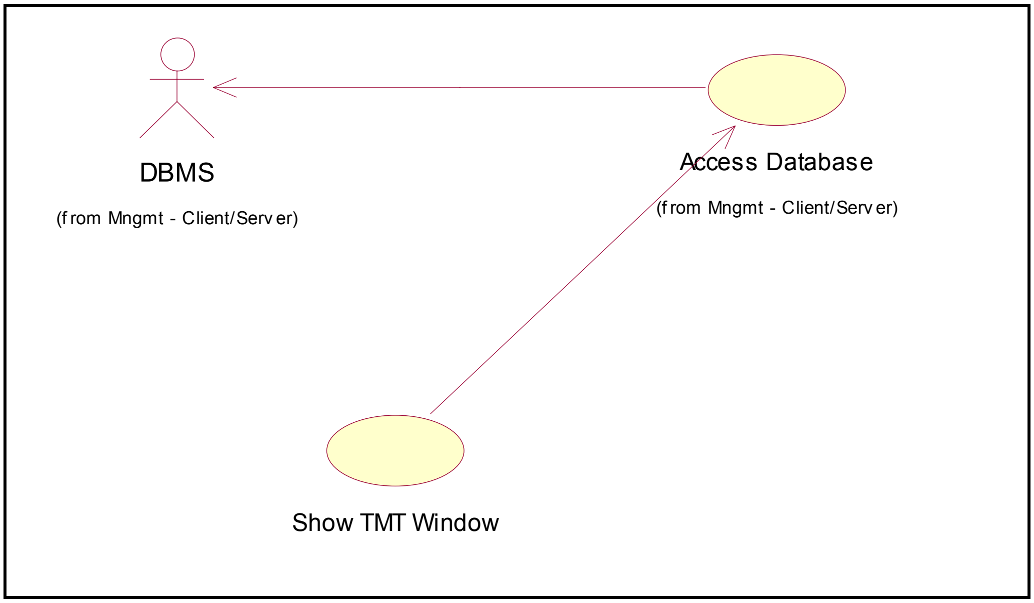

DBMS
(f rom Mngmt - Client/Serv er)
Access Database
(f rom Mngmt - Client/Serv er)
Show TMT Window

Figure 32 : Show TMT Window

8.2 Flow Events

After the username and password have been verified and validated, the SM provides the Manager Client Window to the MCM or provides the Developer Client Window to the DCM. Windows are displayed to the user (Administrator or Developer)

8.3 Interaction Diagrams

- **The SM confirms the Login**
- **The SM provides the appropriate Client Window to the Client Module**
- **The Client Module displays the window to the user.**

8.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case < Show TMT Window>.

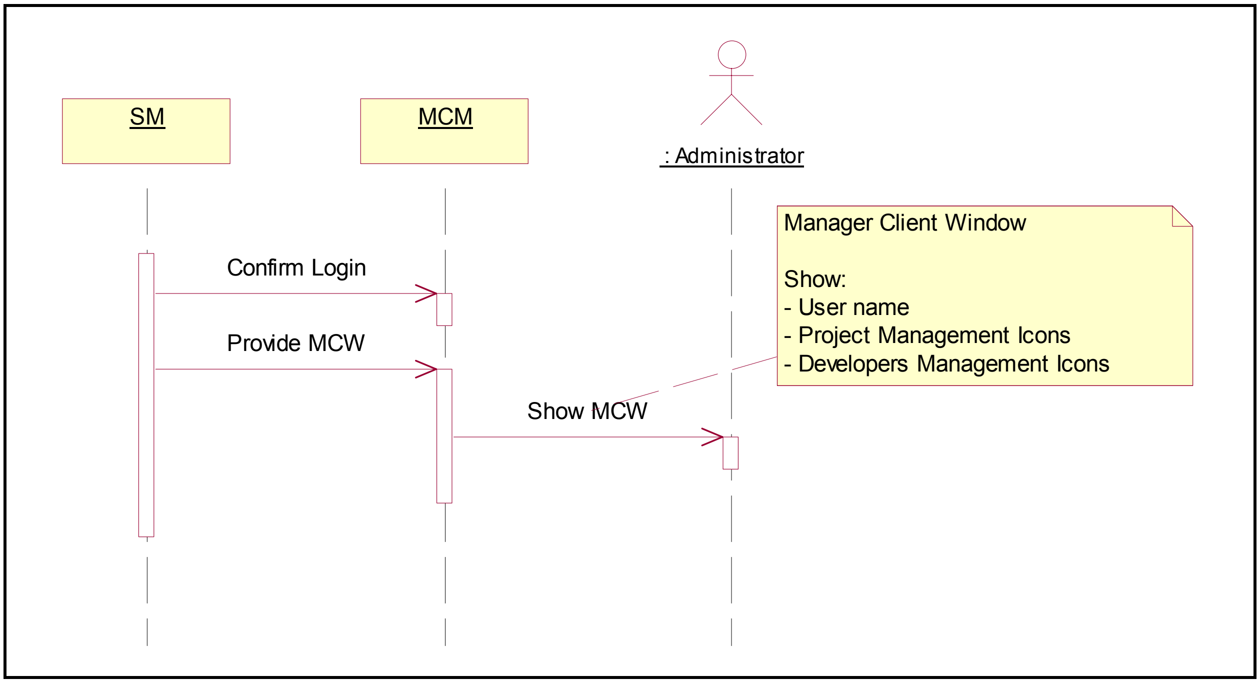


Figure 33 : Sequence Diagram : Show TMT Window (Administrator)

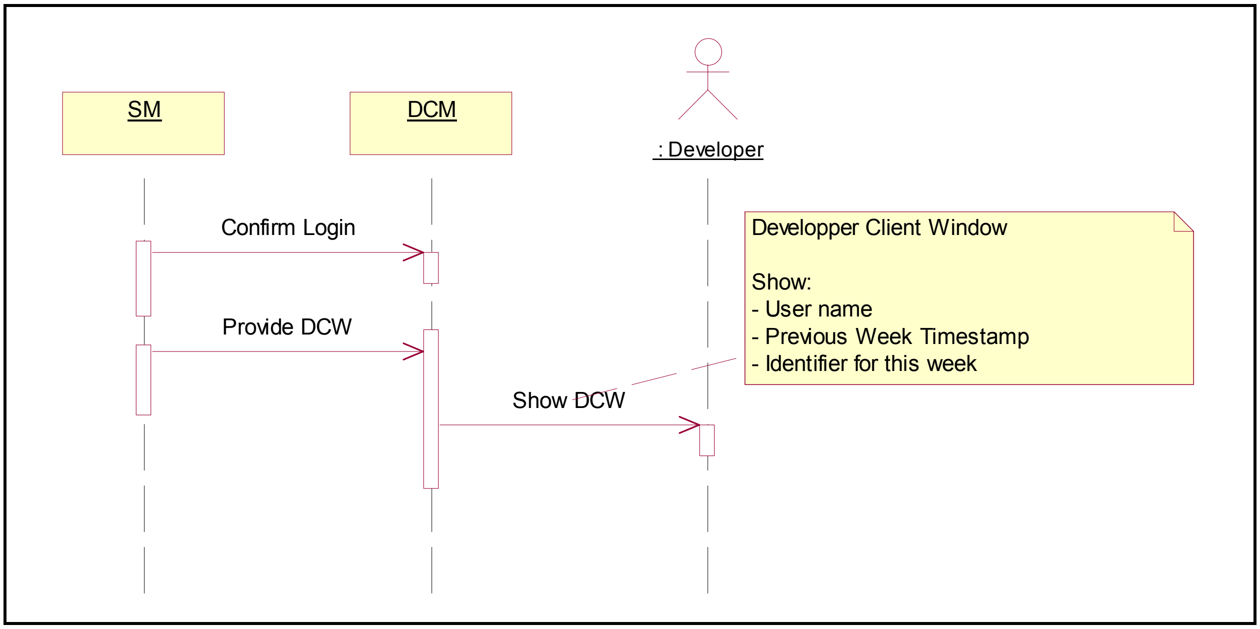


Figure 34 : Sequence Diagram : Show TMT Window (Developer)

8.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <Show TMT Window>.

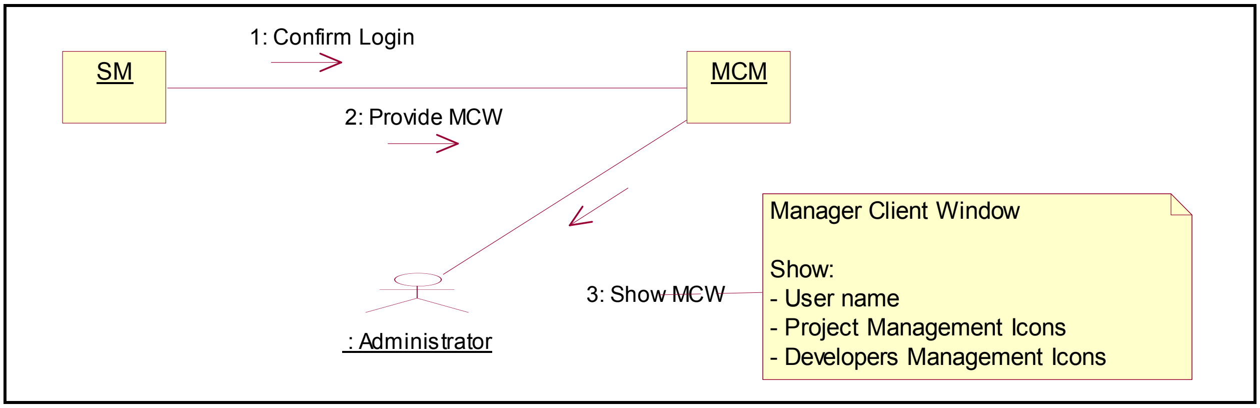


Figure 35 : Collaboration Diagram : Show TMT Window (Administrator)

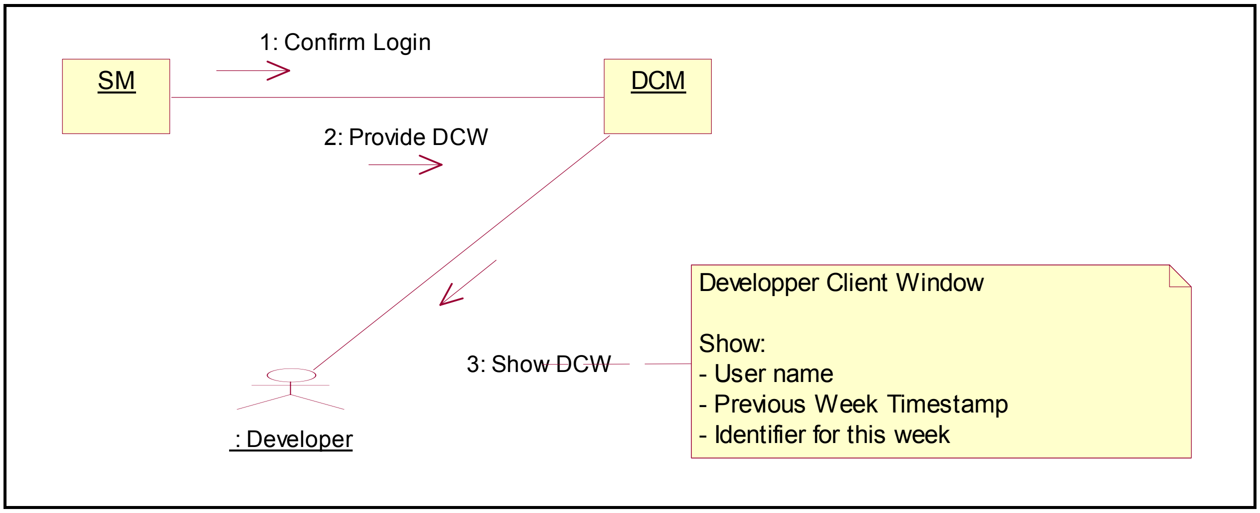


Figure 36 : Collaboration Diagram : Show TMT Window (Developer)

8.4 Participating Objects

The following objects collaborate and define the Use-Case < Show TMT Window> :

SM	**This object interacts with the Database and determines if the user is an Administrator or a Developer**
MCM	**This object interacts with the SM in order to display the TMT client window to the user (Administrator)**
DCM	**This object interacts with the SM in order to display the TMT client window to the user (Developer)**

8.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

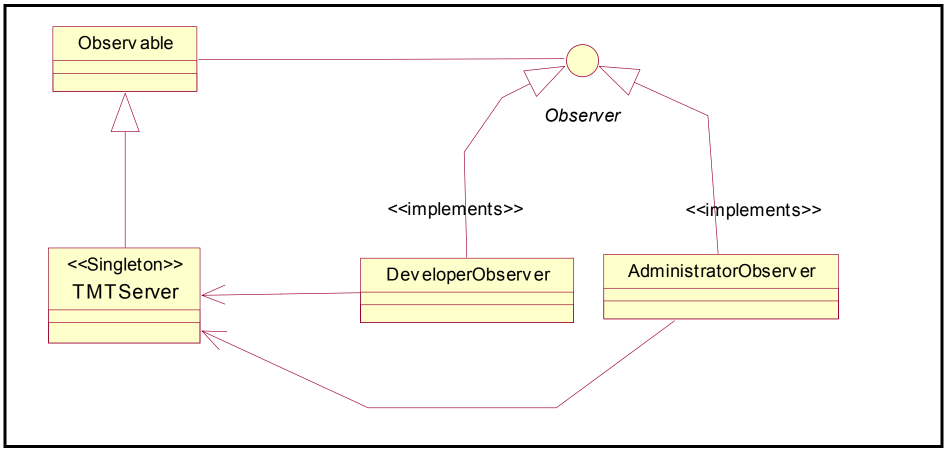


Figure 37 : Object Diagrams : Show TMT Window

9. USE CASE < Close TMT Window >

9.1 Brief Description

This Use-Case defines how the TMT Window is closed upon logging out.

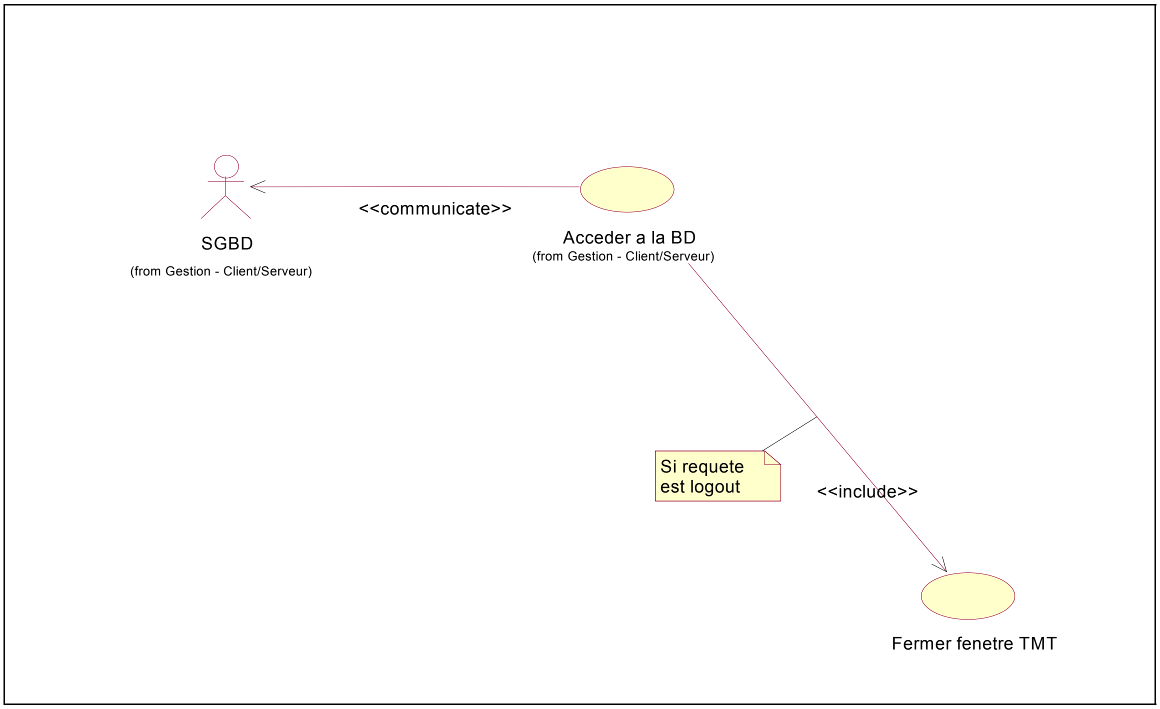


Figure 38 : Close TMT Window

9.2 Flow Events

After a logout query has been accepted and validated by the SM, the Client Modules (DCM or MCM) close the TMT client windows.

9.3 Interaction Diagrams

- **The SM confirms the Logout query**
- **The MCM or DCM closes the MCW or DCW (Manager or Developer Client Window)**

9.3.1 Sequence Diagrams

These Sequence Diagrams show Actors and Objects messages exchange in the Use-Case <Close TMT Window>.

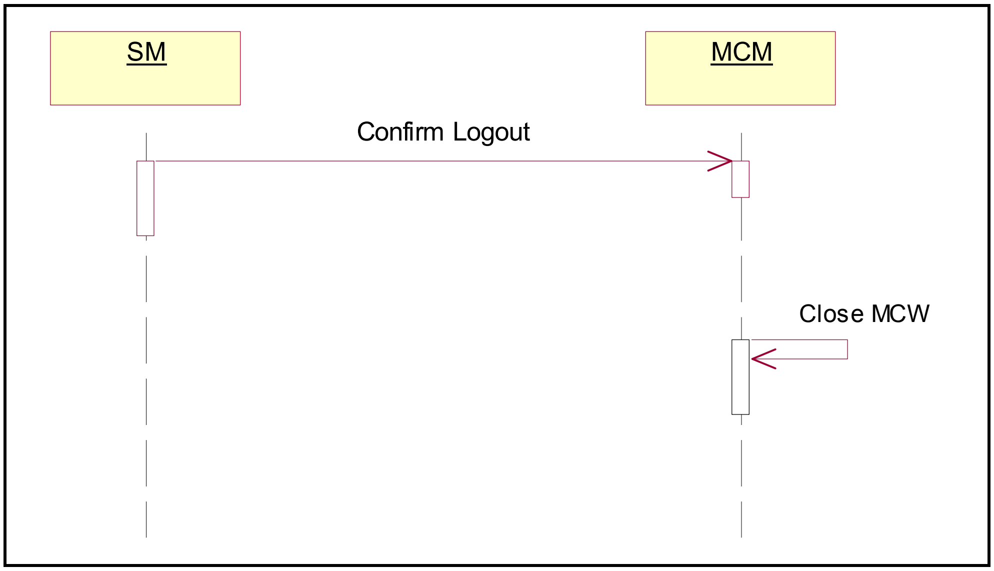


Figure 39 : Sequence Diagram : Close TMT Window (Administrator)

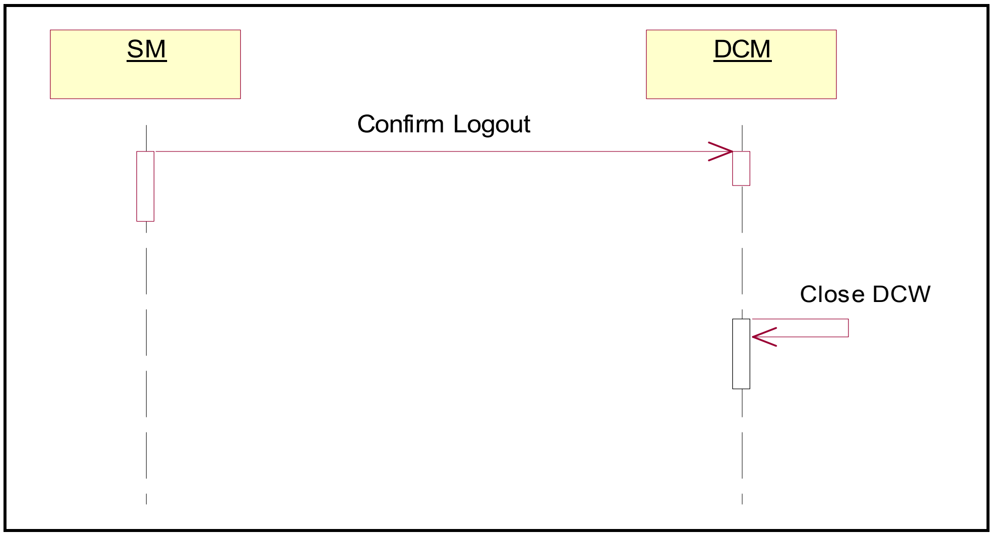


Figure 40 : Sequence Diagram : Close TMT Window (Developer)

9.3.2 Collaboration Diagrams

This Collaboration Diagram shows the static structure of the Use-Case <Close TMT Window>.

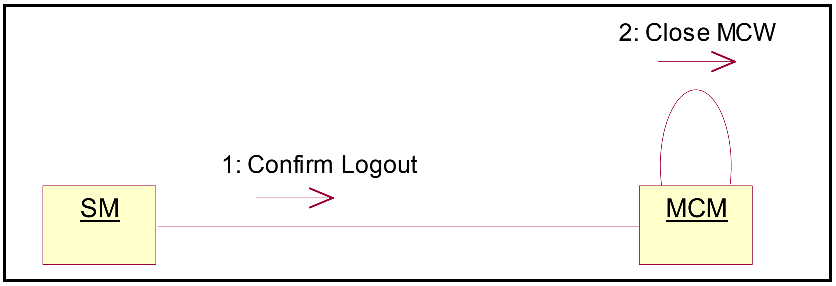


Figure 41 : Collaboration Diagram : Close TMT Window (Administrator)

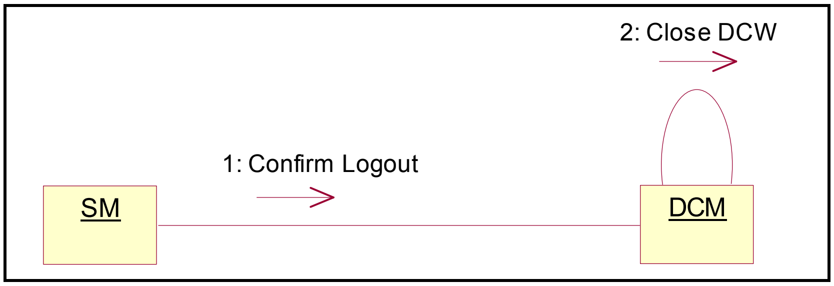


Figure 42 : Collaboration Diagram : Close TMT Window (Developer)

9.4 Participating Objects

The following objects collaborate and define the Use-Case < Close TMT Window> :

SM — **This object interacts with the Database and determines if the user is an Administrator or a Developer**

MCM — **This object interacts with the SM in order to close the TMT client window (Administrator)**

DCM — **This object interacts with the SM in order to close the TMT client window (Developer)**

9.5 Object Diagram

The following Object Diagram shows the relations and constraints between Classes and Objects involved in the Use-Case.

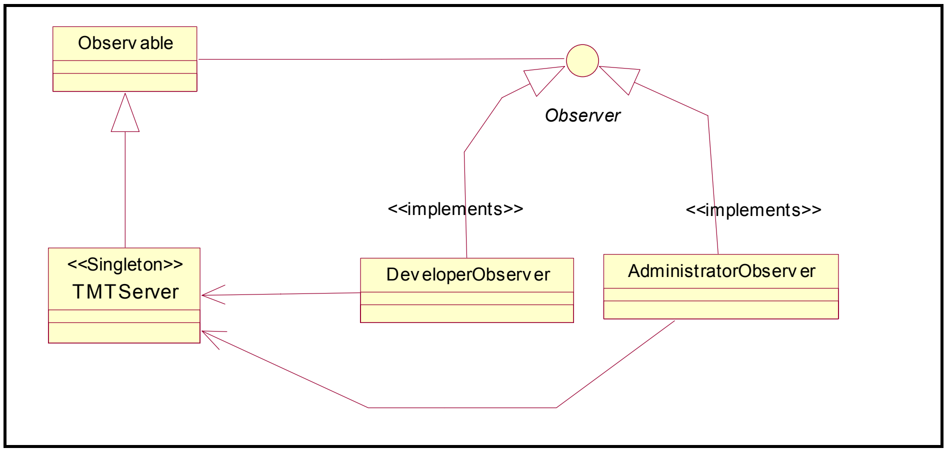


Figure 43 : Object Diagrams : Close TMT Window

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
Defect Report (DF-01)

Version <2.0>

Revision History

Date	Version	Description	Author
<04/03/2001>	<1.0>	Defect entries.	James Prevost
<06/03/2001>	<2.0>	Evaluation of defect details and temporary approbation stamped.	John Lemon

Table of Contents

Defect Report (DF-01)

1. Introduction

1.1 Purpose

The purpose of the Defect Report is to document the unusual events that occur during testing the Time Monitoring Tool system.

1.2 Scope

This Defect Report is linked to Change Request CR-05 and to the Test Cases and Planning documents. Specifically, it refers to Test Case: TC_15

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document

1.4 References

Glossary Document, Glossary, TMT Team 1, École Polytechnique de Montréal, 2001

Test Case Specifications, Test Cases, TMT Team 1, École Polytechnique de Montréal, 2001

Test Plan Document, Test Plan, TMT Team 1, École Polytechnique de Montréal, 2001

1.5 Overview

The following is a detailed report for the defect identified.

2. Defect Report

Defect for Time Monitoring Tool

Created on: <04/03/2001>

Identification

Title: Defect when adding new timestamp

Priority: 3

Status: Opened

Submitted on: 04/03/2001

Defect ID: <DF-01>
Related Change Request ID: <CR-05>

Submitter: James Prevost

Type: Defect

Current Defect

Description: Upon execution of test case TC_15 following test procedure TP_27, the results were not those expected. This test case and the corresponding procedure were expecting a data structure creation in the timestamps database containing all specified fields from the requirements. The procedure was meant to ensure that called methods were executed without any abnormal termination and that results would reflect that execution. Instead, the data structure contained one extra field at the end of it.

Critical Failure: None

Nuisance: The extra field can cause minor database query problems if used. This error has a negligible impact on the remaining tests since this extra field is located at the end of the data structure. However, having an extra field in the structure doesn't follow the stakeholders' requests and specifications.

Source of the defect: Possible explanation: error in creation of the timestamps database tables.

Observation conditions: The test environment is the one described in the corresponding procedure (TP_27). Test was repeated 5 times, with the same result upon every execution.

Proposed Change (Submitter)

Description: Review the database tables, applied modifications, remove unnecessary fields, delete extra fields' data and review timestamps database insertion code block in appropriate file.

Proposed Change (Review Team)

Approval: Under Revision

Reviewed Description: Same as proposed.

Affected Configuration Items	Category	Error Fix
AddJeton.JSP	SCI	Code Level
Timestamps Database	CI	Tables

Resolution

Affected number lines of code: 5

Estimated effort (staff hours): 0,5

Assessment

Tests methods: ▪ Refer to Test Case Document, TC_15	**Test Platforms:** ▪ Refer to Test Plan Document

Test cases: Refer to Test Case Document, TC_15

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool

Test Case

Version <3.0>

Revision History

Date	Version	Description	Author
28/01/2001	1.0	First version	Sandra Lee
05/02/2001	2.0	Added Test Length information	Sandra Lee
10/02/2001	3.0	Added Test information on Timestamps Validation	James Prevost

Preface

The following case study has been modified from its original content. The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Test Section.

Regards,

Unified Process for Education Team

Table of Contents

Test Case

1. Purpose

This document identifies and communicates all conditions to be implemented within the tests. These conditions are mandatory for an acceptable and successful implementation of the final product (following all use-cases and defined constraints).

2. Test case identifier #1

TC_1

2.1 Test item

The test item is the loading of any Client Module. The purpose is to ensure that the system loads and displays correctly the client modules. The DCM must be loaded if the requested connection is made by a Developer. The MCM must be loaded if the requested connection is made by an Administrator. References for this test case are obtained from the following project documentation:

Requirements specification: upedu_ex_srs.pdf
Design specification: upedu_ex_sad.pdf

2.2 Input specifications

The test input is a mouse click on the appropriate icon. The source of the input is the user and it is a real test input

Prerequisite conditions: Once the username and password are validated, the Client Module is loaded.

2.3 Output specifications

If user is a Developer, DCM is loaded. If user is an Administrator, MCM is loaded.

2.3.1 Developer

The system displays a dialog zone with the following properties:

- **Current Week's date**
- **Developer's Username**
- **All fields are clearly identified**
- **Current week's validated and non-validated timestamps are shown**
- **All timestamps information are in a table, having 1 timestamp per row**
- **Each timestamp field is editable**
- **If any, all timestamps to resubmit (Timestamps sent back to the Developer for some reason) are also displayed**

2.3.2 Administrator

The system displays a dialog zone with the following properties:

- **Administrator's Username**
- **Two main menus are displayed: Developers Management and Project Management.**
- **Developers Management menu includes: Add a User, List/Modify/Deactivate and Validate Timestamps.**
- **Project Management menu includes: Export to MS-Excel, Export to MS-Project, Add a Project and Add a Task**

2.4 Environmental needs

2.4.1 Hardware

All test operations need a workstation running under MicroSoft Windows 95/98/NT4/2000.

2.4.2 Software

- **The code for the testing module**
- **The proper version of the WWW browser**

2.4.3 Other

None

2.5 Special procedural requirements

None

2.6 Intercase dependencies

None

3. Test case identifier #2

TC_2

3.1 Test item

All possible Login data to submit. (Usernames and passwords combinations)

3.2 Input specifications

The tester enters the TMT Login window URL in the Browser's address bar.

TEST CASE	USERNAME	PASSWORD
TC_2_01	Valid (Administrator)	Valid (Administrator)
TC_2_02	Valid (Administrator)	Invalid (Administrator)
TC_2_03	Invalid (Administrator)	Valid (Administrator)
TC_2_04	Invalid (Administrator)	Invalid (Administrator)
TC_2_05	Valid (Developer)	Valid (Developer)
TC_2_06	Valid (Developer)	Invalid (Developer)
TC_2_07	Invalid (Developer)	Valid (Developer)
TC_2_08	Invalid (Developer)	Invalid (Developer)

3.3 Output specifications

For <TC_2_01> and <TC_2_05> (Valid entries), if the test is successful, the corresponding TMT Client window is displayed.

For all others, if the test is successful, the corresponding Login error window is displayed.

3.4 Environmental needs

3.4.1 Hardware

All test operations need a workstation running under MicroSoft Windows 95/98/NT4/2000.

3.4.2 Software

- **The code for the testing module**
- **The proper version of the WWW browser**

3.4.3 Other

None

3.5 Special procedural requirements

None

3.6 Intercase dependencies

None

Time Monitoring Tool
Test Evaluation Report

Version <1.0>

Revision History

Date	Version	Description	Author
<28/03/01>	<1.0>	First Version	Robert Latour
		Test Coverage revision	Sabrina Laflamme

Preface

The following case study has been modified from its original content.

The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Test Section.

Regards,

Unified Process for Education Team

Table of Contents

Test Evaluation Report

1. Introduction

1.1 Purpose

The purpose of the Test Evaluation Report is to summarize the results of the testing efforts and provide evaluations based on those results.

1.2 Scope

This Test Evaluation Report describes the results of the tests on Time monitoring Tool system, in terms of test coverage and defect analysis. The tests conducted are described in the Test Plan for the Time Monitoring Tool system. The data used in this evaluation are based on the test results data base and on Test Results Summary documents. This Evaluation Report is to be used for the following:

- assess the acceptability of the performance behavior of the TMT system
- assess the acceptability of the tests
- identify improvements to increase test quality.

1.3 Definitions, Acronyms and Abbreviations

This information may be provided by reference to the project Glossary – upedu_ex_gloss.pdf

1.4 References

The references can be obtained from the documents:

- TMT Test Plan
- TMT Results Report
- TMT Glossary

1.5 Overview

The following sections present and analyze the test results. Some diagrams are added in order to illustrate this analyze.

2. Test Results Summary

The test cases defined in the Test Model for the Time Monitoring Tool were executed following the test strategy as defined in the Test Plan.

Test coverage refers the covering of the use cases and test requirements as defined in the Test Plan. A number of 109 tests were executed. The test coverage was complete.

Package	Executed test	Passed test	Failed test
Total	100%	89.38%	29.42%

Code coverage was not considered as a significant measure of success for the prototype, but for the final product.

The performance tests that involved access to the Time Monitoring Tool system are in the average of the established targets.

3. Requirements-based Test Coverage

The tests to be performed on the prototype are defined in the Test Plan along with their completion criteria. The test coverage results are as follows:

Package	Performed Test Case	Successful Test Cases	Failed Test Cases
Server	25	17	8
Users::Administrator	24	19	5
Users:Developper	28	22	6
Export Data	32	24	8
Total	109	82	27

The area of tests with the highest failure rate was:

- Tests involving access to the Server
- Tests involving export to Excel and MS Project

3.1 Error types and impacts

3.1.1 Data Integrity

Error on test cases outputs.

- *Potential Impact*: High, timestamps validity is important.
- *Relative Impact*: Superior to all other error types

3.1.2 Edge Effect

These errors are caused by using out of range function parameters. If the application cannot control adequately the passed values, an error occurs.

- ***Potential Impact*: Moderate, the application is unstable only in a few extreme cases.**
- ***Relative Impact*: More important than Design divergence errors since there is an instability risk presence.**

3.1.3 Design divergence

These errors are caused by a test case implementation which did not follow initial design.

- ***Potential Impact*: Minor, the application doesn't respond entirely to the fixed requirements which slightly decrease the application ability to fulfill its objectives.**
- ***Relative Impact*: Less important than the Edge effect errors since the application stability is not affected.**

3.2 Error type frequency

The following table shows the error types frequencies for all tests. We can easily see that Design divergence errors have a higher occurrence percentage than other error types.

ERROR TYPE	FREQUENCY
Data Integrity	28%
Edge Effect	12%
Design Divergence	48%

4. Code-based Test Coverage

Approximately, 54% of the code was executed during the testing. It was determined that this coverage was adequate for the prototype tests as all interfaces were thoroughly exercised. Later iterations will require a significantly higher measure for code coverage. The coverage target for the final code has to exceed 90%.

5. Suggested Actions

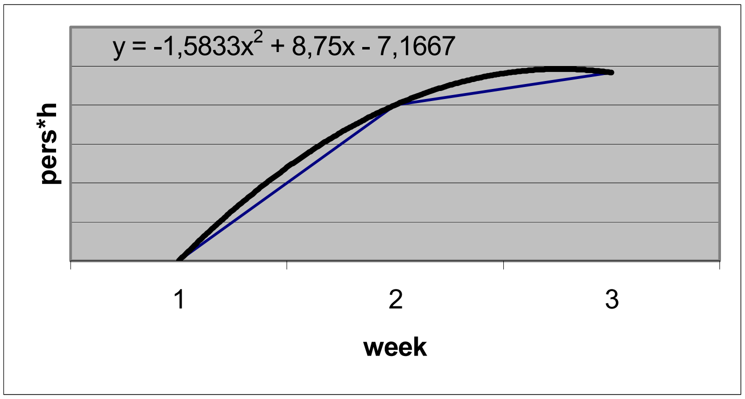
1. Suggested Actions

The recommended actions are as follows:

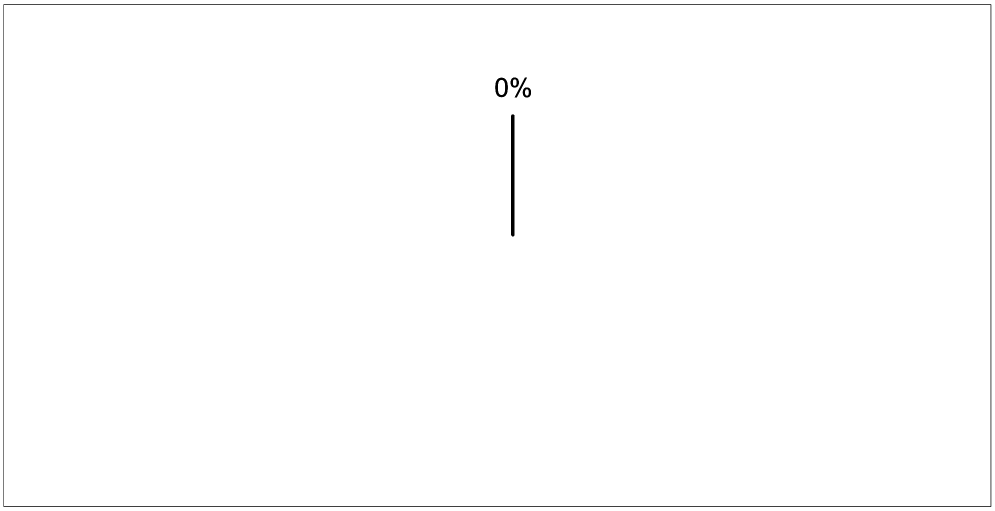
- **Delay start of next iteration pending resolution of Critical Defects.**
- **Design additional tests to further tests.**
- **It is recommended that future iterations include inspections of the all design or code involving external interfaces. These inspections should reduce the number of problems found during Test.**

6. Summary of Activities

The figure below illustrates that the project team follows a pattern with uniformly spread of effort and an intensive work in the second and the third week of the project. The effort pattern could be approximate by a second degree polynomial.



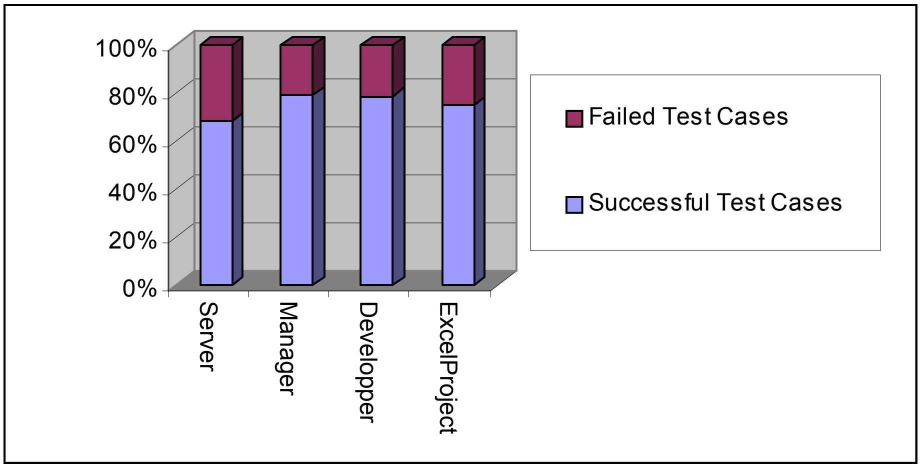
The effort dissipation in these four test activities:



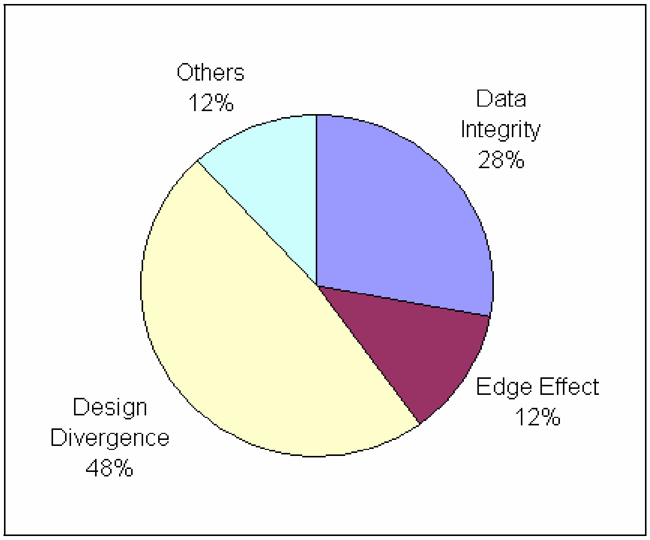
The first iteration is dominated by work on the Requirements discipline. The construction of a prototype required in order to complete the requirements specification shows up as an effort burst during the second and the third week in Test discipline.

Even though the process used is based on the concept of iterations, one can find behaviors that are related to the classic Waterfall software process model, particularly within the engineering disciplines. However, the Test discipline offers a behavior that is very different from the Waterfall model. Much effort is spent on tests at the beginning of the project, on the plan and design test activities.

7. Diagrams



Requirements-based Test Coverage



Error Type Frequency

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool

Test Plan

Version 3.0

Revision History

Date	Version	Description	Author
15/01/2001	1.0	First version	Sandra Lee
26/01/2001	2.0	Revised majors points with Lab attendant	Sandra Lee
01/04/2001	3.0	Final revision and addition of the Timestamps Validation functionality	James Prevost

Table of Contents

Test Plan

1. Introduction

1.1 Purpose

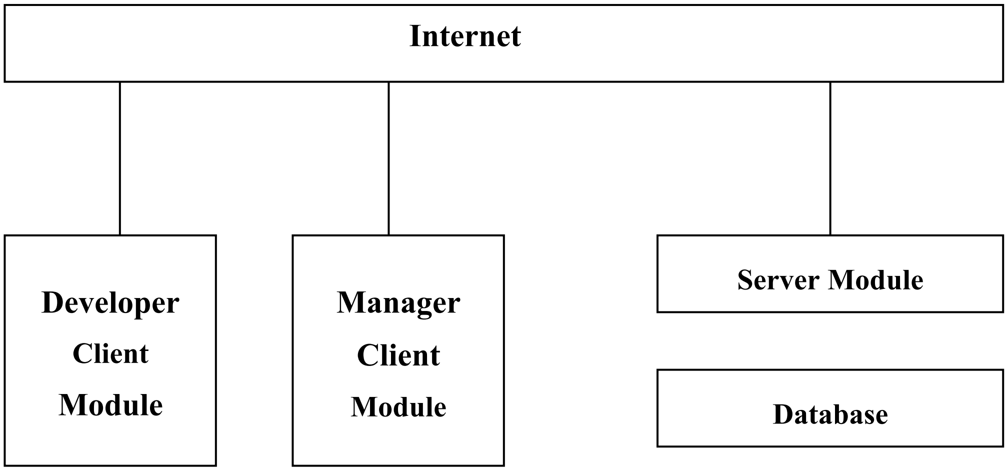
This Test Plan document for the *Time Monitoring Tool* supports the following objectives:

1. **Identify existing project information and the software components that should be tested**
2. **List the recommended Requirements for Test (high level)**
3. **Recommend and describe the testing strategies to be employed**
4. **Identify the required resources and provide an estimate of the test efforts**
5. **List the deliverable elements of the test project**

1.2 Background

The Time Monitoring Tool allows developers working within a defined software development process to record the time spent on the various activities, in a database. The TMT will also allow a manager to derive analyses and produce reports based on the data entered in the system.

The TMT system is a stand-alone tool that is integrated within the organization's Intranet. It consists in four major components: a Developer Client Module, a Server Module, a Database, and a Manager Client Module.



The Developer Client Module allows developers to log onto the TMT system and to record timestamps corresponding to their activities in a convenient way.

The Manager Client Module allows a manager to retrieve timestamp information from the database to produce analyses and reports. The Developer and Manager Client Modules must provide a user interface that is available through a WWW browser.

The Server Module is a daemon accepting connections from Developer and Manager Client Modules and serves as an interface between these modules and a database. The Server Module does not have a user interface other than a command to launch it.

The Database component can be any type of database. It does not have to be developed as a part of the TMT system, as long as the Server Module can interface with an available database system. The Database module does not have a user interface.

1.3 Scope

This Test Plan applies to the unit and integration tests that will be conducted on the Time Monitoring Tool System Release 1.

Unit tests will address functional quality, while system testing will address issues of performance.

The following systems interfaces will be tested:

- Developer Client Module Interface
- Manager Client Module Interface

1.4 Project Identification

The table below identifies the documentation and availability, used for developing the test plan:

Document (and version / date)	Created or Available	Received or Reviewed	Author or Resource	Notes
Requirements Specification	■ Yes □ No	■ Yes □ No		
Use Case Reports	■ Yes □ No	■ Yes □ No		
Design Specifications	■ Yes □ No	■ Yes □ No		
Prototype	■ Yes □ No	■ Yes □ No		
Users Manuals	■ Yes □ No	■ Yes □ No		
Project Plan	■ Yes □ No	■ Yes □ No		

2. Requirements for Test

The listing below identifies those items (use cases, functional requirements, and non-functional requirements) that have been identified as targets for testing. This list represents what will be tested. All interfaces actions; database accesses and internal calculus are also listed.

Functionality	Interface	Database	Calculus
1.1 The user shall be able to load the Developer Client Module within Netscape.	**The DCM is loaded upon a Developer's validated login.**	**Access the *Users* table (SELECT) and verify submitted data (username and password)**	**None**
1.2 The Developer Client Module shall support the logging of users.			
1.2.1 The initial window of the DCM shall contain a field for a user name, a field for a password, and a button labeled login. The password field shall be a "secret" field, which does not display what the user types.	**Auto-verification and validation of the submitted data and ensure that the password is visually encrypted**	**None**	**None**
1.2.2 When a user presses the login button, the DCM shall send a request to the SM to login the user.	**Upon clicking the *LOGIN* button, the DCM sends the right login request to the SM**	**Access the *Users* table (SELECT) and verify submitted data (username and password)**	**None**
1.3 If the logging of a user is successful (see 1.2.2), the DCM shall display the Time Monitoring Window.	**The DCW (Developer Client Window) of the TMT loads.**	**None**	**None**
1.3.1 The Time Monitoring Window shall always display the identifier for the current week.	**Ensure that there is a Date Identifier in the DCM window header.**	**None**	**Get the Current date from the server**
1.3.2 The Time Monitoring Window shall always display the username of the user currently logged in.	**Ensure that there is a Username Identifier in the DCM window header.**	**Get the right username from the data validated while processing the login request (SELECT)**	**None**
1.3.3 The Time Monitoring Window shall display all the validated and non-validated time stamp records previously entered for the current week by the user currently logged in.	**Ensure that the DCM shows correctly all timestamps (all validated and non-validated timestamps of the current week)**	**A database SELECT query retrieves all validated and non-validated timestamps of the current week.**	**Verify the date's day and get the current week's boundaries in terms of dates. Sends this date with the database query as a filter.**
1.3.3.1 The validated and non-validated time stamp records shall be displayed	**Ensure that all data is shown in a dynamic HTML table having one timestamp per row.**	**None**	**None**

in a tabular fashion, with every record being on a separate row.			
1.3.3.2 The fields should be identified with labels. Intuitive and non-confusing abbreviations can be used if necessary.	Ensure that every column's title is appropriate to the data contained within that column.	None	None
1.3.3.3 The fields should be non modifiable to prevent modification of the timestamp records.	Ensure that all non-modifiable fields are preventing the user from any type of edition through his internet browser.	None	None
1.4 The existing Time Monitoring Window timestamps shall be updateable.	Ensure that all modifiable fields allow the user to update the timestamps through his internet browser.	None	None
1.4.1 It shall only be possible to modify the Project, Task, Activity, and Artifacts fields with values that are allowed for the user currently logged in.	Ensure that all available projects (Shown and listed in the DCW) are the user-specific projects.	A database SELECT query retrieves from the *Users* table all active projects from the selected user.	None
1.4.2 It shall only be possible to modify the time fields for every day of the week using a positive numerical value of maximum 24.	Ensure that all dates all editable and shown in the 24:00 format.	None	None
1.4.3 Modifications to the records shall not be synchronized automatically with the server, that is, it should be possible to modify several values and then to click a button or perform another mechanism to update the server.	All modifications must be saved when a user submits the timestamps forms (new timestamps or corrected timestamps)	INSERT and UPDATE queries to the database in the *Timestamps* table to save all modifications	Ensure that all data has been modified.
1.5 The Time Monitoring Window shall support the entry of new timestamps.	The user must be able to add new timestamps.	None	None
1.5.1 It shall be possible to add new timestamps according to predefined fields.	Ensure that upon adding a new timestamp, all fields have a default value.	A SELECT query in the Users table ensures that the default values for this user are shown.	None
1.5.2 It shall be possible to create up to 100 new timestamps for a given user	Ensure that at least 100 timestamps can be added to the database within the same week for a given user.	None	None

and a given current week.			
1.5.3 All new timestamps should come with the predefined selections for a given user.	Ensure that upon adding a new timestamp, all fields have a default value.	A SELECT query in the Users table ensures that the default values for this user are shown.	None
1.5.4 All time entry shall be in hours and minutes H:mn.	All time fields must be shown in the HH:mm format. The interface scripts ensure that more complex formats (e.g.: retrieved from the database) are downsized to this format.	None	None
1.5.5 All empty time fields shall be at 0.	By default all time fields in the forms are set to the 00:00 default value	None	None
1.6 The Time Monitoring Window shall support entry of timestamps.	The DCW interface shows an “Add new timestamps” form which is operational.	None	None
1.6.1 All fields of a timestamp shall have predefined values for the logged in user.	Ensure that upon adding a new timestamp, all fields have a default value.	A SELECT query in the Users table ensures that the default values for this user are shown.	None
1.6.2 Project, Task, Activity, Artifact and at least one time field must be filled out before the record is sent to the database.	The DCW interface ensures that the Project field, the Task field, the Activity field and the Time field must be filled before submitting the form	None	Internal scripting to the interface verifies the non-filled fields and generates error messages, if needed.
1.6.3 Time entry could be in duration or taxi mode.	The interface allows the Taxi mode activation and the simple duration calculus.	None	When taxi mode is activated, Internal scripting sets a timer until the Timestamp’s End Time is entered. Then the timer is set off.
1.6.4 Time Monitoring Window shall provide a running clock for the taxi mode.	Ensure that the Taxi mode timer is visible when this mode is selected.	None	Simple timer calculus: current time minus start time. Always refreshes the timer display.

1.7 The Time Monitoring Window shall provide cumulative totals.	**Ensure that the statistical module is showing all available totals (by column, by day, by project)**	**A SELECT query retrieves the current week's timestamps in the *Timestamps* table**	**A sum of all available timestamps data**
1.7.1 Each column day of the current week shall have the total number of hours recorded.	**Ensure that every day of the current week has its total number of hours shown**	**A SELECT query retrieves the current week's timestamps in the *Timestamps* table**	**A sum of every day's timestamps data**
1.7.2 Each project and task line of the current week shall have the total number of hours recorded.	**Ensure that every task and project of the current week have their total number of hours shown**	**A SELECT query retrieves the current week's timestamps in the *Timestamps* table**	**Horizontal sum**

Functionality	Interface	Database	Calculus
2.1 The manager shall be able to load the Manager Client Module within Netscape.	**The MCM is loaded upon an Administrator's validated login.**	**Access the *Users* table (SELECT) and verify submitted data (username and password)**	**None**
2.2 The Manager Client Module shall support the logging of managers.			
2.2.1 The initial window of the MCM shall contain a field for a user name, a field for a password, and a button labeled login. The password field shall be a "secret" field, which does not display what the user types.	**Auto-verification and validation of the submitted data and ensure that the password is visually encrypted**	**None**	**None**
2.2.2 When a user presses the login button, the MCM shall send a request to the SM to login the user.	**Upon clicking the *LOGIN* button, the MCM sends the right login request to the SM**	**Access the *Users* table (SELECT) and verify submitted data (username and password)**	**None**
If the logging of a user is successful (see 2.2.2), the MCM shall display the Manager Client Window.	**The MCW (Manager Client Window) of the TMT loads.**	**None**	**None**
2.3.1 The Manager Client Window shall always display the username of the manager currently logged in.	**Ensure that there is a Date Identifier in the MCM window header.**	**None**	**Get the Current date from the server**

2.3.2 The Manager Client Window shall display two groups of icons, which are for the user management and the project management.	**Ensure that both menus are shown and that all icons and options are present for each menu.**	**None**	**None**
2.4 Manager client Window shall support user's management.	**See 2.4.x**	**None**	**None**
2.4.1 The user management icons shall include: add a user, display/modify/disable users, and validate timestamps.	**Ensure that all User Management functionalities are shown in the appropriate menu.**	**None**	**None**
2.4.2 Clicking add user icons shall display a fill in form for adding a user to the project.	**Ensure that clicking the *Add User* option loads the Adding a new user form web page.**	**None**	**None**
2.4.2.1 Add user form shall enable the recording of the user id, the projects, the user's supervisor identification, and the selection of predefined fields for this user.	**Ensure that the Adding a new user form contains and shows all the needed fields and that default values are present.**	**A SELECT query to the database, retrieving the available project and supervisors**	**Ensure that all fields are filled correctly.**
2.4.2.2 Add user form shall be validated for completeness before being sent to the Server Module.	**Externally query the database (NOT in TMT, for example a Database management software) and verify that the same values are retrieved**	**Same SELECT statement as above to the database, retrieving the available project and supervisors**	**None**
2.4.3 Clicking Display all users icon should display the list of all users in alphabetical order with the identification of the projects that they are involved in and their supervisor.	**Ensure that upon clicking the *List Users* option, the list is sorted alphabetically and that all mentioned fields are shown beside each user entry.**	**A SELECT query in the *Users* table to retrieve all users data using a simple ORDER BY clause**	**None**
2.4.3.1 Display all users could provide a list ordered alphabetically by last name of the user or by project or by supervisor.	**Ensure that the interface has multiple sort order options available and operational (by name, project, or supervisor)**	**A SELECT query in the *Users* table to retrieve all users data using a simple ORDER BY clause**	**None**

2.4.3.2 By selecting a user, his profile can be changed or disabled.	**Ensure that the status of a given user is changed.**	**A SELECT query in the *Users* table to retrieve all users data using a simple ORDER BY clause**	**None**
2.4.4 Clicking validate timestamps should display the list of all timestamps **Records submit by users.**	**Ensure that upon click on the *Validate Timestamps* option, list contains only the new (non-validated) timestamps submitted by the TMT users.**	**A SELECT query in the *Timestamps* database.**	**None**
2.4.4.1 The timestamp records shall be displayed in a tabular fashion, with every record being on a separate row.	**Ensure that the interface's dynamic table shows each timestamp on a separate row.**	**None**	**None**
2.4.4.2 The fields should be identified with labels. Intuitive and non-confusing **Abbreviations can be used if necessary.**	**Ensure that the dynamic columns have a header row with appropriate and clear titles.**	**None**	**None**
2.4.4.3 The manager can validate a timestamp record, refuse a timestamp record or leave timestamp record non-validated.	**Ensure that all timestamp validation options are present and operational. Ensure that the interface allows the timestamp's status modification**	**An UPDATE query in the *Timestamps* table modifies the status**	**None**
2.4.4.4 The validated timestamps records should be inserted in the database.	**Ensure that all validated timestamps are not shown in the validation table.**	**A SELECT query filters the *Timestamps* database and retrieves the right timestamps**	**None**
2.4.4.5 The refused timestamps records should be returned to the user to allow correction.	**Ensure that all rejected (sent back to user for correction) timestamps are not shown in the validation table.**	**A SELECT query filters the *Timestamps* database and retrieves the right timestamps**	**None**
2.4.4.6 The non-validated timestamps record should be stayed in the table to allow manager to validate or refuse a timestamp record another time.	**Ensure that, at any time, all non-validated timestamps are shown in the validation table regardless of the different actions performed on the table data.**	**A SELECT query filters the *Timestamps* database and retrieves the right timestamps**	**None**
2.5 Manager client Window shall support project's management.	**See 2.5.x**	**None**	**None**
2.5.1 The project's management icon shall include the add project, add task, MS EXCEL and MS Project icons.	**Ensure that all Project Management functionalities are shown in the appropriate menu.**	**None**	**None**

2.5.2 Clicking the MS EXCEL icons shall display a form to identify the project, users, week's data and the EXCEL file where the corresponding data from the database should be transferred.	Ensure that upon clicking on the *Export to EXCEL* option, the export form is shown with the appropriate fields and data selection options. Ensure that the exportation form includes a functional file selection input for the exportation destination.	A SELECT query in all TMT database tables to retrieve all exportable data.	None
2.5.3 Clicking the MS-Project icons shall display a form to identify the project; users, week's data and the MS-Project file where the corresponding data from the database should be transferred.	Ensure that upon clicking on the *Export to MSPROJECT option*, the export form is shown with the appropriate fields and data selection options. Ensure that the exportation form includes a functional file selection input for the exportation destination.	A SELECT query in all TMT database tables to retrieve all exportable data.	None
2.5.4 Clicking add project icon shall display a form to identify the new project to add.	Ensure that upon clicking the *Add Project* option, the appropriate form is loaded.	None	None
2.5.5 Clicking the add task icon shall display a form to add a new task in a project selected.	Ensure that upon clicking the *Add Task* option, the appropriate form is loaded.	None	None

Functionality	Test to execute
3.1 The Server module shall be the only intermediate between the two client modules and the database.	None test A simple architecture verification and validation will do
3.2 The Server Module shall receive all the requests and format the pages.	Start with basic SELECT queries to the database and generate a simple dynamic HTML table in a web page. If that result is successful, all further requests can be based on this model. See 3.6 & 3.8 below.
3.3 The Server Module shall accept all connections from developers and manager client modules.	Overdrive the SM will multiple connections made simultaneously and ensure that all made requests have been accepted.
3.4 Upon log in request from the DCM the server module shall produce the Time Monitoring Window as specified in 1.2.	Ensure (login as a Developer) that the DCM is loaded. See 1.2 above.
3.5 Upon request for updating time stamp records from the DCM the server module shall update the database to reflect the new state of the DCM.	When a developer modifies timestamps data, the SM sends an UPDATE or an INSERT query to the database following simple SQL rules. The database should accept the queries and internally perform the modifications.

3.6 The Server Module shall validate and execute all requests coming from the DCM.	For all functionalities specified in requirements 2.1 to 2.5.4.1, the SM executes the requested actions. Use a multi-connections environment and query all MCM actions on each connection. All actions should be accepted. The SM returns the requested action, interacts with the database and returns all database actions details, then gets the results and returns these results also.
3.7 Upon log in request from the MCM the server module shall produce the Manager Client Window as specified in 2.2.	Ensure (login as an Administrator) that the MCM is loaded. See 2.2 above.
3.8 The Server Module shall validate and execute all requests coming from the MCM.	For all functionalities specified in requirements 2.1 to 2.5.4.1, the SM executes the requested actions. Use a multi-connections environment and query all MCM actions on each connection. All actions should be accepted. The SM returns the requested action, interacts with the database and returns all database actions details, then gets the results and returns these results also.
3.9 The Server Module shall display the TMT status.	In both DCM and MCM use the SM; query the database with wrong entries which should create an abnormal query termination. The status of the TMT should change and an error message will be displayed.
3.9.1 Any error of execution, communication, validation or else shall be identified and appropriate comment display.	Launch any action or query then, interrupt internet connection on the server side for a short time. Remove database files or tables.
3.9.2 The server module shall try to recovery from most common errors.	None

3. Test Strategy

The Test Strategy presents the recommended approach to the testing the target-of-test. The main considerations for the test strategy are the techniques to be used and the criterion for knowing when the testing is completed.

3.1 Testing Types

3.1.1 Function Testing

Test Objective:	Ensure proper target-of-test functionality, including navigation, data entry, processing, and retrieval.
Technique:	Execute each use case, use case flow, or function, using valid and invalid data, to verify the following: • The expected results occur when valid data is used. • The appropriate error / warning messages are displayed when invalid data is used. • Each business rule is properly applied.
Completion Criteria:	• All planned tests have been executed. • All identified defects have been addressed.
Special Considerations:	

3.1.2 User Interface Testing

Test Objective:	Verify the following: • Navigation through the target-of-test properly reflects requirements, including window to window, field to field, and use of access methods • Window objects and characteristics, such as menus, size, position, state, and focus conform to standards.
Technique:	Create / modify tests for each window to verify proper navigation and object states for each application window and objects.
Completion Criteria:	Each window successfully verified to remain consistent with benchmark version or within acceptable standard
Special Considerations:	Not all properties for custom and third party objects can be accessed.

3.1.3 Data and Database Integrity Testing

Test Objective:	**Ensure Database access methods and processes function properly and without data corruption.**
Technique:	**Invoke each database access method and process, seeding each with valid and invalid data (or requests for data).** **Inspect the database to ensure the data has been populated as intended, all database events occurred properly, or review the returned data to ensure that the correct data was retrieved (for the correct reasons)**
Completion Criteria:	**All database access methods and processes function as designed and without any data corruption.**
Special Considerations:	**Testing may require a driver to enter or modify data directly in the database.** **Processes should be invoked manually.**

3.1.4 Performance Profiling

Test Objective:	**Verify performance behaviors for designated transactions under the following conditions:** **- normal anticipated workload** **- anticipated worse case workload**
Technique:	**Use Test Procedures developed for Function Cycle Testing.** **Modify data files to increase the number of transactions, or the scripts to increase the number of iterations each transaction occurs.**
Completion Criteria:	**Single Transaction / single user: Successful completion of the test scripts without any failures and within the expected / required time allocation (per transaction)** **Multiple transactions / multiple users: Successful completion of the test scripts without any failures and within acceptable time allocation.**
Special Considerations:	**Methods that can be used:** • **“Drive transactions” directly to the server, usually in the form of SQL calls.** • **Use multiple physical clients, each running test scripts to place a load on the system.** **Performance testing should be performed on a dedicated machine or at a dedicated time. This permits full control and accurate measurement.** **The databases used for Performance testing should be either actual size, or scaled equally.**

3.1.5 Load Testing

Test Objective:	**Verify performance behaviors time for designated transactions or business cases under varying workload conditions.**
Technique:	**Use tests developed for Function Cycle Testing.** **Modify data files (to increase the number of transactions) or the tests to increase the number of times each transaction occurs.**
Completion Criteria:	**Multiple transactions / multiple users: Successful completion of the tests without any failures and within acceptable time allocation.**
Special Considerations:	**Load testing should be performed on a dedicated machine or at a dedicated time. This permits full control and accurate measurement.** **The databases used for load testing should be either actual size, or scaled equally.**

3.1.6 Stress Testing

None

3.1.7 Volume Testing

Test Objective:	**Verify that the target-of-test successfully functions under the following high volume scenarios:** • **Maximum (actual or physically capable) number of clients connected (or simulated) all performing the same, worst case (performance) for an extended period.** • **Maximum database size has been reached (actual or scaled) and multiple queries / report transactions are executed simultaneously.**
Technique:	**Use tests developed for Performance Profiling or Load Testing.** **Multiple clients should be used, either running the same tests or complementary tests to produce the worst case transaction volume / mix (see stress test above) for an extended period.** **Maximum database size is created (actual, scaled, or filled with representative data) and multiple clients used to run queries / report transactions simultaneously for extended periods.**
Completion Criteria:	**All planned tests have been executed and specified system limits are reached / exceeded without the software or software failing.**
Special Considerations:	**What period of time would be considered an acceptable time for high volume conditions (as noted above)?**

3.1.8 Security and Access Control Testing

Test Objective:	**Application-level Security: Verify that an actor can access only those functions / data for which their user type is provided permissions.** **System-level Security: Verify that only those actors with access to the system and application(s) are permitted to access them.**
Technique:	**Application-level: Identify and list each actor type and the functions / data each type has permissions for.** **Create tests for each actor type and verify each permission by creating transactions specific to each user actor.** **Modify user type and re-run tests for same users. In each case verify those additional functions / data are correctly available or denied.** **System-level Access (see special considerations below)**
Completion Criteria:	**For each known actor type, the appropriate function / data are available and all transactions function as expected and run in prior function tests**
Special Considerations:	**Access to the system must be reviewed / discussed with the appropriate network or systems administrator. This testing may not be required as it maybe a function of network or systems administration.**

3.1.9 Failover / Recovery Testing

None

3.1.10 Configuration Testing

Test Objective:	**Verify that the target-of-test functions properly on the required hardware / software configurations.**
Technique:	**Use Function Test scripts** **Open / close various non-target-of-test related software, such as the Microsoft applications, Excel and MS Project, either as part of the test or prior to the start of the test.** **Execute selected transactions to simulate actor's interacting with the target-of-test and the non-target-of-test software** **Repeat the above process, minimizing the available conventional memory on the client.**
Completion Criteria:	**For each combination of the target-of-test and non-target-of-test software, all transactions are successfully completed without failure.**

Special Considerations:	What non-target-of-test software is needed is available, accessible on the desktop? What applications are typically used? What data are the applications running (i.e. large spreadsheet opened in Excel, large document in MS Project). The entire systems, NetWare, network servers, databases, etc. should also be documented as part of this test.

3.1.11 Installation Testing

None

3.2 Tools

The following tools will be employed for this project:

	Tool
Project Management	Microsoft Project

4. Resources

The resource for the *Time Monitoring Tool* test effort is the development team.

4.1 Workers

This table shows the staffing assumptions for the project.

Human Resources		
Worker	Resources Recommended (number of workers allocated full-time)	Specific Responsibilities/Comments
Test Manager	1	Provides management oversight Responsibilities: • Provide technical direction • Acquire appropriate resources • Management reporting

Test Designer	1	Identifies, prioritizes, and implements test cases Responsibilities: • Generate test plan • Generate test model • Evaluate effectiveness of test effort
Tester	1	Executes the tests Responsibilities: • Execute tests • Log results • Recover from errors • Document change requests
Designer	1	Identifies and defines the operations, attributes, and associations of the test classes Responsibilities: • Identifies and defines the test class(es) • Identifies and defines the test packages
Implementer	1	Implements and unit tests the test classes and test packages Responsibilities: • Creates the test classes and packages implemented in the test model.

4.2 System

All system tests will be executed on (at the most) 2 or 3 different computers: one from the CISCO laboratory and personal developers' computer to manage DCM and MCM accesses. The Server Module will be mainly tested on a computer located in the Ecole Polytechnique School (C-211).

The server tests stations must have the following softwares installed and properly configured:

JAVA 2 SDK v1.3

Apache v1.3

Tomcat v3.2.1 (Apache)

mySQL Server v3.23

mySQL JDBC driver

The main test station for the Server Module is a PIII-900Mhz, 256MB RAM running under Windows 2000. The computer is operational at anytime and available for testing purposes 24 hours a day and 7 days a week. The IP address is: 132.207.114.212 (static)

In order to make all Server Module tests more reliable, multiple tests will be executed at different traffic times: by day, by night, by morning, by the peak hour.

5. Project Milestones

Milestone Task	Effort	Start Date	End Date
Plan Test	10h	10/01/01	26/01/01
Design Test	15h	28/01/01	12/02/01
Implement Test	15h	12/02/01	20/02/01
Execute Test	30h	22/02/01	20/03/01
Evaluate Test	10h	24/03/01	30/03/01

6. Deliverables

6.1 Test Model

The Test Model will define all the Test Cases and will reference the test procedures and test scripts which are associated with each test case.

6.2 Test Results

For each test executed, a test result form will be created. This shall include the name or ID of the test, the use case or supplemental specification to which the test relates, the date of the test, the ID of the tester, required pre-test conditions, and results of the test.

6.3 Test Evaluation Report

A final evaluation of test activities will be presented.

Time Monitoring Tool
Test Results

Version <1.0>

Revision History

Date	Version	Description	Author
24/03/01	1.0	First results of the test activity.	Sabrina Laflamme

Preface

The following case study has been modified from its original content.

The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Test Section.

Regards,

Unified Process for Education Team

Table of Contents

Test Results

1. Introduction

1.1 Purpose

To summarize the results of the testing activities in Time Monitoring Tool system development and to provide data for evaluations based on these results.

1.2 Scope

All test results are captured during the execution of tests. The test results are used as input for evaluating test and for calculating the key measures of test.

1.3 Definitions, Acronyms, and Abbreviations

Please reference to the project's Glossary – upedu_ex_gloss.pdf

1.4 References

1. TMT - Glossary
2. TMT - Use Case Specification
3. TMT – Test Plan

1.5 Overview

The following sections describe the variances of the executed tests by respect to the previous test documents. A summary of the test results are presented.

2. Variances

All executed tests respected Test Plan.

Important notice: 41 test cases were planned for the Server package but only 25 were executed. The remaining test cases were rejected because other test groups had fewer test cases planned, which caused a disproportion.

3. Summary of results

109 test cases executed (Total).
27 errors detected.
Results were grouped by package.

Test Results – Server Package	
Test case	Status
CT_1 – CT_12	Succeeded
CT_13 – CT_16	Failed
CT_17 – CT_21	Succeeded
CT_22 – CT_25	Failed

Test Results - Package Users::Administrator	
Test case	Status
CT_1 – CT_6	Succeeded
CT_7 – CT_9	Failed
CT_10 – CT_14	Succeeded
CT_15 – CT_16	Failed
CT_17 – CT_24	Succeeded

Test Results - Package Users::Developer	
Test case	Status
CT_1 – CT_12	Succeeded
CT_13 – CT_18	Failed
CT_19 – CT_28	Succeeded

Test Results - Package Export Data	
Test case	Status
CT_1 – CT_16	Succeeded
CT_17 – CT_20	Failed
CT_20 – CT_28	Succeeded
CT_29 – CT_32	Failed

All detailed information about failed tests is provided in corresponding Defect Report document.

Time Monitoring Tool
Change Request Report < CR 01>

Version <1.0>

Revision History

Date	Version	Description	Author
02/02/2001	1.0	First version	Sandra Lee

Table of Contents

Change Request Report < CR 01>

1. Introduction

1.1 Purpose

This specific change request is used to track enhancement request for a change to the product. It provides a record of decisions and, due to its assessment process, ensures that change impacts are understood across the project.

1.2 Scope

This change request report is related to the MCM and was issued after a review with the project's lab attendant.

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document

1.4 References

Glossary Document, Glossary, Team 1, École Polytechnique de Montréal, 2001

1.5 Overview

The following report explains the change requested in details.

2. Change Request Report

Change Request for Time Monitoring Tool **Created on: <02/02/2001>**

Identification

Title: New attribute "Task" for a project.	**Priority:** Medium	**Status:** Closed
	Submitted on: 02/02/2001	
	Change Request ID: <CR-01**>**	
Submitter: James Provost	**Type: <**Enhancement**>**	

Current Problem

Description: Currently all tasks are not related to any specific project. Instead general tasks are available for selections. These tasks are manually added to the database by the database administrator or can be added through the Manager Client-Module.

Critical Failure: N/A

Nuisance: When creating a new project, it is difficult to predict what tasks will be needed for that specific project. Therefore, having default or general tasks in the system is not appropriate.

Source of the Problem: Misunderstanding of the SRS

Enhancement: The tasks should be affiliated to a specific project. A project should be created and then tasks are created for that specific project. By default, a project does not have any tasks.

Observation conditions: During a review of a prototype version of TMT with the Lab Attendant, while accessing to the Add Project functionality in the Manager Client-Module.

Proposed Change (Submitter)

Description: Update the relations in the database tables between project and tasks so we have a *one-to-many* relation (Project => Tasks). Remove default task selection in the MCM when adding a project (HTML/JSP simple implementation). Add 3 new HTML elements in the New Project Window: New tasks Text Input with an add button, and a simple dynamic HTML table showing all project-specific related tasks. Implement new AddTask button with the proper JSP/SQL query (considering the Project ID and the tasks primary key [ProjectID+TaskID]. Conduct another query for the simple dynamic table (Simple SELECT of all tasks with the proper primary key related to the project the manager is looking at.

Proposed Change (Review Team)

Approval: Accepted

Reviewed Description: Nothing to add.

Affected Configuration Items	Category	Enhancement/New Requirements/Other
Design Model	SCI	Redesign of AddTask Class
Use-Case Realizations	SCI	Review flow events of Add a Task Use-Case and all related collaboration diagrams
Software Architecture Document	SCI	Update references to Design Model and Use-Case Realizations

TMT Database	SCI	Update table relations between a project and a task for cascading queries to be performed correctly
Code – AddTache.jsp	SCI	Implement new functionality and test queries

Resolution

Resolution shall be implemented following the submitter proposed change.

Estimated effort (staff hours): 5 hours, 1 worker (100% used)

Change Review Team Disposition

Changes approved and accepted on: 02/05/2001	**By:** John Lemon
Changes implemented on: 02/20/2001	**By:** Robert Latour

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Configuration Management Plan

Version <3.0>

Revision History

Date	Version	Description	Author
15/01/2001	1.0	First Version	Robert Latour
10/02/2001	2.0	Lab attendant's required modifications applied.	Robert Latour
29/03/2001	3.0	Major revision: adapted content to the new modules created.	Robert Latour

Table of Content

Figures

Configuration Management Plan

1. Introduction

1.1 Purpose

This document describes all Configuration and Change Control Management (CCM) activities to be performed during the course of the project. It details the schedule of activities, the assigned responsibilities, and the required resources, including staff, tools, and computer facilities.

1.2 Scope

This document is complementary to the Software Development Plan. For more information on the Organization, the responsibilities, the infrastructure, the milestones and the resources refer to the Software Development Plan. (See References)

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document (See References)

1.4 References

Glossary Document, Glossary, Team 1 TMT, École Polytechnique de Montréal, 2001

Software Development Plan Document, Software Development Plan, Team 1 TMT, École Polytechnique de Montréal, 2001

Course Web Page INF4302: http://www.cours.polymtl.ca/INF4302/

UPEDU: http://www. upedu.org/

1.5 Overview

At first, the document explains the software configuration management environment then goes on with the configuration management program (including Configuration Identification, Configuration and Change Control and Configuration Status and Accounting). More information is provided on the milestones strategies, the training and resources used and the external development environment.

2. Software Configuration Management

2.1 Organization, Responsibilities, and Interfaces

All team members are participating to the CM activities. Although Robert Latour has been assign the "official" Configuration and Change manager.

2.2 Tools, Environment, and Infrastructure

Since most of the project members will work remotely (mainly at home), a FTP Server will be used. All environment files will be available at any time at the following address:

Host: los.dhs.org

Port: 21 (standard)

Login: INF4302

Password: TiTaNiC

Moreover, the Configuration Manager will save a backup copy on the space allowed to the team in the course lab.

3. The Configuration Management Program

3.1 Configuration Identification

3.1.1 Identification Methods

The product artifacts are to be named following this rule:

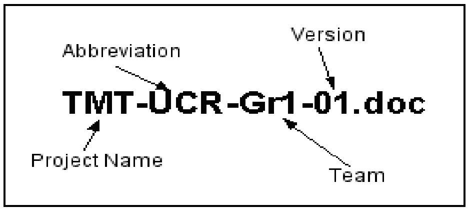


Figure 1 : Artifacts' Identification Method

3.1.1.1 Artifacts Abbreviations Policies

Artifacts' abbreviation must always have 3 characters. Also, always use lowercase characters for non-starting letters. Here are some examples:

Single-word artifacts:	**Glossary (Glo)**
Two-word artifacts:	**Supplementary Specifications (SuS)**
More-than-2-words artifacts:	**Use-Case Realization (UCR)**

3.1.1.2 Artifacts to produce

Here is a list of all artifacts' abbreviation:

Artifact	Abbreviation
Analysis Classes	AnC
Analysis Classes Diagram	ACD
Build	Bui
Change Request Report	CRR
Component	Com
Configuration Management Plan	CMP
Design Class	DeC
Design Class Diagram	DCD
Design Model	DeM
Glossary	Glo
Implementation Model	ImM
Implementation Model Document	IMD
Iteration Plan #	IP#
Iteration Plan Diagram	IPD
Measurement Plan	MeP
Prototype	Pro
Review Record	ReR
Software Development Plan	SDP

Supplementary Specification	SuS
Test Cases	TeC
Test Components	TsC
Test Evaluation Report	TER
Test Plan	TeP
Test Results	TeR
Timestamps	Tim
Use-Case Diagram	UCD
Use-Case Realization	UCR
Use-Case Specifications	UCS
Work Order	WoO

3.1.2 Product directory structure (Project Repository)

The project repository used is the FTP site describe under Section 2.2.

Project Referential is defined in section 3.1.2.3.

3.1.2.1 Workspaces

Development Workspaces: All development workspaces are private to each team member. This project team will use each member's own workstation as a development workspace. Access to the integration workspace is made via FTP technology.

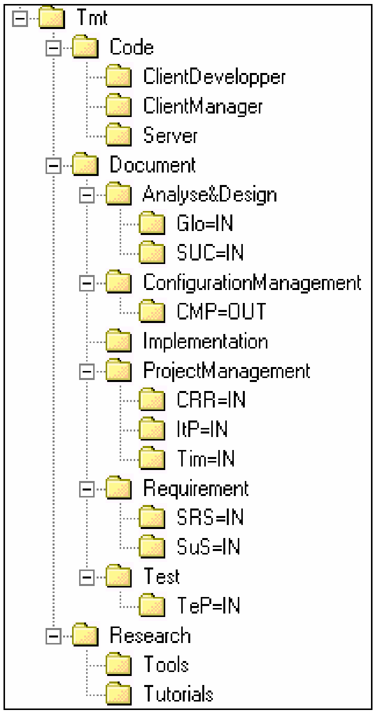
Integration Workspace: Since the project repository is hosted on a FTP site, this site is acting as the Integration Workspace too. Baselining policies are defined in Section 3.1.3. All project artifacts are accessible through the FTP site. This is where the overall product is being built.

3.1.2.2 Product directory naming conventions

All directories are named after the section they represent. If the section is more than one-word long, the words are concatenated

3.1.2.3 Base Referential

The Referential follows this structure (This is a sample):



```
Tmt
    Code
        ClientDevelopper
        ClientManager
        Server
    Document
        Analyse&Design
            Glo=IN
            SUC=IN
        ConfigurationManagement
            CMP=OUT
        Implementation
        ProjectManagement
            CRR=IN
            ItP=IN
            Tim=IN
        Requirement
            SRS=IN
            SuS=IN
        Test
            TeP=IN
    Research
        Tools
        Tutorials
```

Figure 2 : Referential directory structure

3.1.3 Project Baselines

To ensure documents integrity, special folders like « Artifact Abbreviation »=IN or « Artifact Abbreviation »=OUT will be used. IN and OUT flags are the document's status. For example, if a team member wants to work on the artifact Software Requirements Specifications (SRS), he renames the artifact's folder from SRS=IN to SRS=OUT. This indicates all other team member that this artifact is currently being worked on or modified and will have to be reviewed by the project team before putting back the folder to SRS=IN.

After each iteration, a build is produced and put in the project repository.

3.1.3.1 Special folder for current code

Testers need the current code files to be available at anytime. A special folder, named "Current", is used for than matter. This folder is a sub-folder of folder "Code" in the project directory structure. All coded files are contained within this folder, uncompressed.

3.2 Configuration and Change Control

3.2.1 Change Request Processing and Approval

The only team member authorized to add documents to the project referential is the configuration manager and only this member can modify the content (structure) of the referential. Robert Latour has been assigned to this function and applies the following steps when submitting an artifact on the referential:

- **Quickly revise the document**
- **Put the document in the "ToRevise" folder (External to the referential)**
- **Advise all team member that a Document Review is required (mostly using email or telephone). Reviews will take place mostly on Fridays between 11:00 am and 2:30 pm.**
- **Document modifications are iterative and distributed among team members.**
- **When the document is accepted by all reviewers, Robert Latour transfers the document in its associated folder in the project referential using the FTP site.**

3.3 Configuration Status Accounting

3.3.1 Project Media Storage and Release Process

Backup storage is made through another FTP site, this one located in the course lab itself: the Software Engineering Research lab. The configuration manager ensures that all TMT versions are stored and backed up on this FTP site, using the following login information:

Host : 132.207.28.54

Port: 21 (standard)

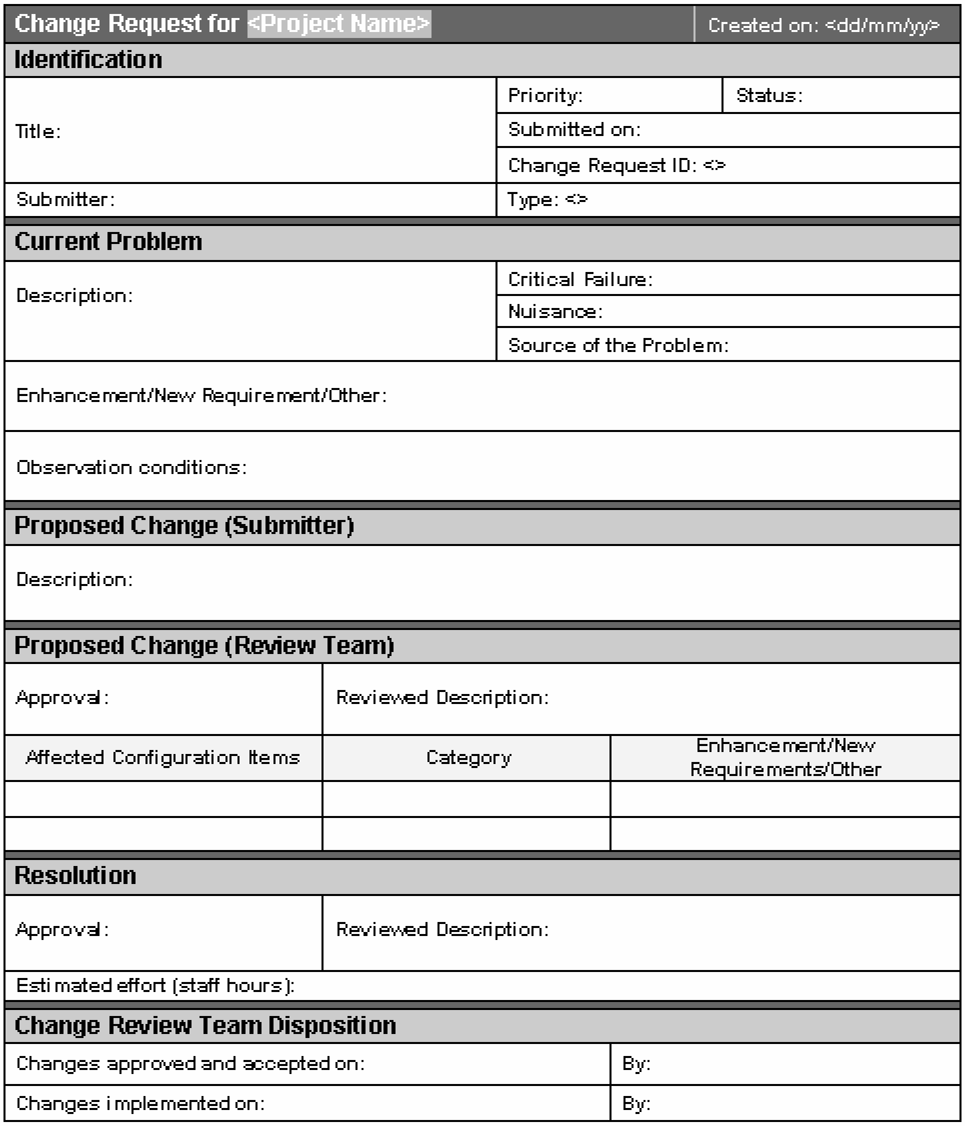
Login: INF4302_gr1

Password: TiTaNiC

Security backups are made every Friday, after the reviews between 5:30 pm and 6:00 pm. This backup version of the project will also be used by the project instructor for follow-ups.

3.3.2 Reports and Audits

Only one report is used in the TMT project: The Change Request Report. The template used for the report is the following:



Change Request for <Project Name> Created on: <dd/mm/yy>

Identification

Title:

Priority: Status:

Submitted on:

Change Request ID: <>

Submitter:

Type: <>

Current Problem

Description:

Critical Failure:

Nuisance:

Source of the Problem:

Enhancement/New Requirement/Other:

Observation conditions:

Proposed Change (Submitter)

Description:

Proposed Change (Review Team)

Approval:

Reviewed Description:

Affected Configuration Items	Category	Enhancement/New Requirements/Other

Resolution

Approval:

Reviewed Description:

Estimated effort (staff hours):

Change Review Team Disposition

Changes approved and accepted on: By:

Changes implemented on: By:

4. Milestones

The CM Plan should be updated after each iteration. See Software Development Plan Document for more information on minor and major milestones.

5. Training and Resources

Software tools: MicroSoft Office, Any FTP Client, Any Email Client, availability of any used templates and documentation.

No Training is required.

6. External development environment

Refer to Section 2.2.

ÉCOLE POLYTECHNIQUE MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
Iteration Plan <Iteration 4>

Version <1.0>

Revision History

Date	Version	Description	Author
<19/03/2001>	1.0	First Version	John Lemon

Table of Contents

Iteration Plan <Iteration 4>

1. Introduction

1.1 Purpose

In this fourth iteration, the team's mission is to develop a final version of the TMT system. All TMT functionalities must be implemented and operational. All graphical user interfaces must be in their final state, except for little modifications that might be applied during the transition phase.

1.2 Scope

This plan targets tasks and activities assignments especially in relation with artifacts production and client modules finalization and realization. This plan offers a vision on how these tasks and activities will be assigned among team members and what roles are involved during this iteration:

Implementer

The implementer participates to the following activities: implement all graphical user interfaces for the client modules (Manager Client Windows and Developer Client Windows),

Related Artifacts are: Code (Build) and Graphical User Interface Prototype Document.

Integrator

The integrator participates to the following activities: Integrate system and subsystems (Both client modules and graphical interfaces)

Related Artifacts are: Implementation Model (Document and Model itself)

Tester

The tester participates to the following activities: Plan tests for client modules functionalities and for all graphical interfaces interactions.

Related Artifacts are: Test Plan, Test Results, and Test Case

Reviewer

The reviewer participates to the following activities: Review code, Review Project Planning.

Related Artifacts are: Review Record.

Project manager

The project manager participates to the following activities: Plan phases and iterations, Develop iteration plan, Schedule and Assign work.

Related Artifacts are: Software Development Plan, Iteration Plan, Gantt Diagram, Work Order, Team's internal database including all team members' timestamps.

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document (See References)

1.4 References

- Glossary Document, Glossary, TMT Team 1, École Polytechnique de Montréal, 2001
- Software Development Plan document, Software Development Plan, TMT Team 1, École Polytechnique de Montréal, 2001
- Gantt Diagram – All 4 iterations, Gantt Diagram MSProject File, TMT Team 1, École Polytechnique de Montréal, 2001
- Works Orders from Iteration 4, Work Orders Documents, TMT Team 1, École Polytechnique de Montréal, 2001
- Test Plan Document, Test Plan, TMT Team 1, École Polytechnique de Montréal, 2001
- Test Cases Specifications, Test Case, TMT Team 1, École Polytechnique de Montréal, 2001
- Test Results, Test Results, TMT Team 1, École Polytechnique de Montréal, 2001
- Implementation Model (UML Model), Implementation Model (Model), TMT Team 1, École Polytechnique de Montréal, 2001
- Implementation Model (Document), Implementation Model Document, TMT Team 1, École Polytechnique de Montréal, 2001
- Graphical User Interfaces Document, User Interface Prototype Document, TMT Team 1, École Polytechnique de Montréal, 2001
- The UPEDU http://www.upedu.org
- The course web page http://www.cours.polymtl.ca/inf4302/

1.5 Overview

The document presents the planning for the iteration and all resources needs.

2. Plan

Refer to Gantt Diagrams Document – *upedu_ex_allit.pdf*

3. Resources

3.1 Human resources

- ***Project's team:*** **John Lemon, James Prevost, Sandra Lee, Robert Latour, Sabrina Laflamme.**
- **All course members: professors and lab assistants.**

3.2 Software resources

- **Rational Rose Enterprise 2000**
- **MS-Project 2000**
- **Borland J-Builder 4**
- **MSOffice 2000 Premium**
- **Netscape Navigator v4.75+**
- **Internet Explorer v5.5+**
- **FTP Client**
- **Email Client**

3.3 Hardware resources

- **The course labs**
- **Personal computers**

4. Use Cases

Iteration-Related Use-Cases:

- **Login**
- **Logout**
- **Validate Query**
- **Execute Query**
- **Show TMT Status**
- **Identify Errors**
- **Show TMT Window**
- **Close TMT Window**
- **Add Timestamp**
- **Validate Timestamp**
- **Add User**
- **List User**
- **Modify User**
- **Deactivate User**
- **Export to EXCEL**
- **Export to MSPROJECT**

ÉCOLE POLYTECHNIQUE MONTRÉAL

Le génie sans frontières

Time Monitoring Tool Measurement Plan

Version 2.0

Revision History

Date	Version	Description	Author
<22/01/01>	1.0	**First metrics entries**	**John Lemon**
	1.1	**Developed metrics**	**John Lemon**
	1.2	**Reviewed and added metrics**	**John Lemon**
	2.0	**Page layout and global review (corrections)**	**John Lemon**

Preface

The following case study has been modified from its original content.

The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages). Note that The Measurement Plan may physically be part of the Software Development Plan if the metrics program is a simple one.

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Project Management Section.

Regards,

Unified Process for Education Team

Table of Contents

Measurement Plan

1. Introduction

1.1 Purpose

The purpose of the Software Measurement Plan is to specify the primitive metrics which should be collected and the metrics which should be computed during the project. For the project progress monitoring, a set of project goals should be specified.

1.2 Scope

This Software Measurement Plan defines a simple metrics program, including the measurement goals, the associated metrics, and the primitive metrics to be collected in the project, in order to monitor its progress. The details of the individual activities will be described in the Iteration Plans. The plans as outlined in this document are based upon the software requirements as defined in the SRS Document.

1.3 Definitions, Acronyms, and Abbreviations

See the Project Glossary

1.4 References

For the Software Measurement Plan, the list of referenced artifacts includes:

- **SRS**
- **Software Development Plan**
- **Iteration Plans**
- **Project Measurements**
- **Glossary**

1.5 Overview

This *Software Measurement Plan* contains the following information:

Measurement Goals: **The goals of the measurement program relative to the project in terms of achievement, improvement, and quality.**

Metrics: **The metrics that are to be synthesized at regular intervals on the project to support the goals**

Primitive metrics: **The primitive metrics that are collected, automatically or manually, to compute the metrics**

2. Measurement Program Goals

There are essentially two approaches that can be followed to ensure end product quality, one being assurance of the process by which a product is developed, and the other being the evaluation of the quality of the product.

- **the Process — the sequence of activities invoked to produce the software product (and other artifacts)**
- **the Product — the artifacts of the process, including software, documents and models**

The goals of the Measurement Program relative to the Time Monitoring Tool project are the following:

- **Evaluate:**
  - **the quality of the product**
  - **The quality of code**
  - **The quality of documents and models**
- **Assure**
  - **a documented follow-up of the project**
  - **the progress of tasks according to estimates**
  - **the state of the activities**
- **Evaluate**
  - **the impact of the changes to the requirements**
  - **tasks implied by the requirements**
  - **The realization level of the requirements**
- **Facilitate**
  - **the planning and the estimate of new projects**
  - **The relative importance of disciplines**
  - **The importance of the activities inside disciplines**

3. Metrics

3.1 Product

The products are the code, the documents and the models. Some specific characteristics and their associated metrics are listed in the following table:

	Characteristics	Metrics
Code	Size	SLOC
	Size under configuration control	SLOCc
	Depth of inheritance tree	DIT
	Adaptability	E/N, average effort per SCO
Documents	Size	Page count
	Effort	Staff-time units for production, change and repair
	Volatility	Number of changed pages
	Traceability	Not measured directly: judgment made through review, using a checklist
Models	Size	Number of Use Cases Number of actors Number of classes Number of packages Methods per class Attributes per class Number of Test Cases

3.2 Process

To completely characterize the process, measurements should be made at the lowest level of formally planned activity. Activities will be planned by the Project Manager. A record should then be kept of actual values over time and any updated estimates that are made.

Metrics	Comments
Effort	Staff effort units (staff-hours)
Duration	Elapsed time for the activity
Output	Artifacts and their size and quantity (note this will include defects as an output of test activities)
Effort application	The way effort is spent during the performance of the planned activities

3.3 Metrics

Name	WMC
Definition	The Weighted Methods per Class (WMC) is a count of the methods implemented within a class.
Goals	The number of methods involved is a predictor of how much time and effort is required to develop and maintain the class. The larger the number of methods in a class, the greater the potential impact on children since children will inherit all the methods defined in a class. Classes with large numbers of methods are likely to be more application specific, limiting the possibility of reuse. This metric measures usability and reusability.
Analysis Procedure	The Weighted Methods per Class will be counted by the software design tool – Rational Rose. The data will be collected al the end of the corresponding activities for the Design Class Model artifact. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

Name	Duration
Definition	The duration will be calculated based on start and end time: Duration = End_Time – Start_Time The Duration will be calculated in minutes. The Start and the End Time were approximated to the nearest quarter of an hour.
Goals	The Duration metric is being collected to permit the effort calculation. Effort measurement will be used for product measurement and for process measurement.
Analysis Procedure	All team members will use the same template for the effort recording information, presented in Annex / point 5.1. The tool used to register the time spent on each activity is presented in Annex / point 5.2. The data will be saved in the Measurement Data Base.
Responsibilities	Each team member will be responsible for filling out his records capturing the time spent on a specific activity, once the activity is ended. One team member will be responsible for collecting every week his team-mates' completed records and for integrating the data into the Measurement Database.

4. Primitive Metrics

The following metrics will be collected manually to compute the metrics:

Name	SLOC
Definition	Source Line of Code
Collection procedure	SLOC will be count by the programming environment. The data will be collected at the end of each iteration. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

Name	SLOCc
Definition	Source Line of Code for the current baseline
Collection procedure	SLOC will be counted by the programming environment. The data will be collected each time the base line is changed. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

Name	Number of Use Cases
Definition	Number of Use Cases in the Use Case Model
Collection procedure	The use cases will be counted by the software design tool – Rational Rose. The data will be collected al the end of the corresponding activities for the Use Case Model artifact. The data will be saved in the Measurement Data Base.
Responsibilities	Software Engineering Team

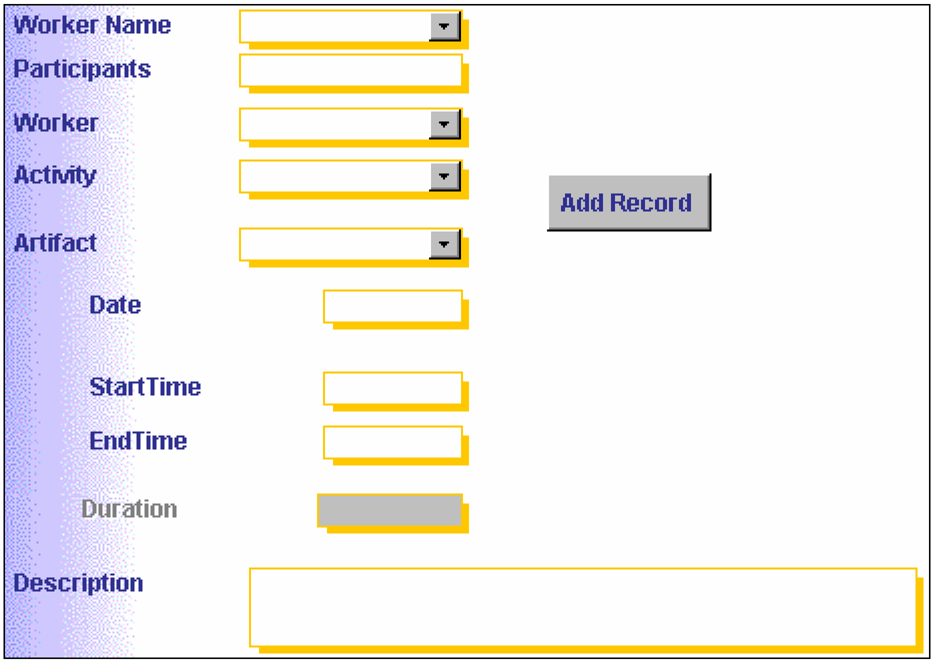
5. Annexes

5.1 Format of effort recording information

Worker Name	Worker	Activity	Artifact	Date	Start Time	End Time	Duration

For each record, the students inserted a short description of the entered data.

5.2 Tool used to register the time



ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
Review Record <16-02-01>

Version <1.0>

Revision History

Date	Version	Description	Author
<16/02/2001>	<1.0>	First Version	Sabrina Laflamme

Table of Contents

Review Record <16-02-01>

1. Introduction

1.1 Purpose

This document captures the results of the review of several Time Monitoring Tool Project artifacts (User Interface Prototype, Use-Case Specifications, Use-Case Realizations, Use-Case Diagrams).

1.2 Scope

This document encloses the detailed information of an inspection review on the artifacts stated above. This review was necessary before the end of Iteration #2.

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document. (See References)

1.4 References

Glossary Document, Glossary, TMT Team 1, École Polytechnique de Montréal, 2001

1.5 Overview

This document identifies the naming convention of the review record ID. Further, the review record itself is detailed.

2. Review Record ID convention

The naming convention used for the review records ID is the date the review occurred. Thus, IDs are <DD-MM-YY) of the review.

3. Review Record Report

Review Record for Time Monitoring Tool	Created on: <16/02/2001>
Identification	
Review Record ID: <16-02-01>	**Type: <**Artifacts Inspection Review**>**
General Information	
Artifacts Reviewed: ▪ User Interfaces Prototype ▪ Use-Case Specifications ▪ Use-Case Realizations ▪ Use-Case Diagrams	**Objectives of the Review:** ▪ See if these artifacts suit the needs of the project conductors ▪ See if these artifacts follow the right conventions

Review Participants	
Participant name	**Role during the meeting**
Sabrina Laflamme	Recorder
Sandra Lee	Moderator
Robert Latour	Presenter
James Prevost	Reviewer
John Lemon	Reviewer

Review Schedule and Location	
Location: École Polytechnique de Montréal, CISCO Lab.	**Review date:** <16/02/01>
	Review time: <13:30>
Reviewed artifacts Publication Schedule	
User Interface Prototype	**Publication date:** <17/02/01>
Use-Case Specifications	**Publication date:** <17/02/01>
Use-Case Realizations	**Publication date:** <17/02/01>
Use-Case Diagrams	**Publication date:** <17/02/01>

Problems identified	
User Interface Prototype:	There is 2 different windows in the DCM for add/modify timestamps. Export buttons are missing in MCM – Project Management. In MCM – Add User, there shouldn't be *Default Worker/Task/Activity* fields. Also project should be selected from a Select Box, not radio buttons. The word **Delete** is inappropriate.
All Others:	<<*extend*>> arrows are inversed and Netscape Actor is ambiguous

Recommendations

User Interface Prototype: In the DCM, have only one window for modifying and adding new timestamps. In the MCM – Project Management, add export (Excel and MSProject) buttons. In the MCM – User Management, remove the *worker/task/activity* fields and replace the *Radio Selection* for projects with a *Select Box.* Replace the word "**Delete**" with "**Deactivate**".

All Others: Inverse the <<*extend*>> arrows and change the *Netscape* Actor.

Action Items Status

User-Interface Prototype: Raise Work Orders
All Others: Continue work

Issues for Consideration by the Project Manager

None

Follow Up Review

No other review is necessary for these modifications.

Record on Effort

Preparation: 0,5 hour per team member.
Review Conducting: 0,5 hour

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Risk List

Version <1.0>

Revision History

Date	Version	Description	Author
<14/01/01>	<1.0>	First Version: Risks identified at the inception phase	John Lemon

Table of Contents

Risk List

1. Introduction

1.1 Purpose

This document is designed to capture the perceived risks to the success of the project. It identifies, in decreasing order of priority, the events that could lead to a significant negative outcome. It serves as a focal point for project activities and is the basis around which iterations are organized

1.2 Scope

The Risk List is maintained throughout the project. It is created early in the Inception phase, and is continually updated as new risks are uncovered and existing risks are mitigated or retired. At a minimum, it is revisited at the end of each iteration, as the iteration is assessed.

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document, See References

1.4 References

Glossary Document, Glossary, TMT Team 1, École Polytechnique de Montréal, 2001

Iteration Plan Document, Iteration Plan <Iteration 1>, TMT Team 1, École Polytechnique de Montréal, 2001

Software Development Plan Document, Software Development Plan, TMT Team 1, École Polytechnique de Montréal, 2001

1.5 Overview

Risks are organized in a table (Section 2) along with their *magnitude*, *description*, *impacts*, *indicator* and *mitigation/contingency strategies*.

2. Risks

The following risk list uses these conventions:

Magnitude: **Risks are ranked from 1 to 10. 1 is the lowest risk and 10 is the highest risk. Ranking is based upon the criticality of the risk and the probability of the risk occurring.**

Description: **Brief description of the identified risk**

Impact: **C – Critical (Affects all project functionalities and baselines.)**

H – High (Affects stakeholders needs and major product functionalities)

M – Medium (These risks are subject to contingency but most of the times, a mitigation plan will be established in order to avoid the risk.)

L – Low (Generally these are risks for which *Risk Acceptance* strategies will be held or quick mitigation plan will be implemented. Usually the team will decide to live with the risk as a contingency)

Indicator: **Metrics concerned.**

Mitigation/Contingency: **Plan to live with or avoid/transfer the risk**

<RSK-01> - Browsers Incompatibility

Magnitude	Description	Impact	Indicator	Mitigation Strategy / Contingency Plan
4	Incompatibility with internet browsers and specific configurations on client machines.	H	Adaptability	Use Cross-Browsers Technologies and Cascading Styles Sheets Level 1 & 2 Standards.

<RSK-02> - Process Knowledge

Magnitude	Description	Impact	Indicator	Mitigation Strategy / Contingency Plan
3	The development team is relatively inexperienced with the Unified Process for EDUcation (UPEDU). This could lead to lower efficiency and poorer product quality	M	Quality, Completeness, traceability	Establish 'process mentors' who can assist the team in understanding the process and the development activities. Ensure all Design and Code is inspected.

<RSK-03> - Member departure

Magnitude	Description	Impact	Indicator	Mitigation Strategy / Contingency Plan
2	One of the team members quit (for any reason).	C	Effort Application	Review Baselines and major milestones as quick as possible. Review Functional Requirements Classification and Reassign work.

ÉCOLE POLYTECHNIQUE MONTRÉAL

Le génie sans frontières

Time Monitoring Tool
Software Development Plan

Version <1.1>

Revision History

Date	Version	Description	Author
10/01/01	1.0	First Draft	Sabrina Laflamme
12/01/01	1.1	Completion of Document	John Lemon

Preface

The following case study has been modified from its original content.

The case study is meant to be used as a starting point to help you understand how to use the artifact. Thus, information has been shrunk to avoid navigating an enormous document (in size and pages).

You can also refer to the related template (in HTML format or WORD format) in the UPEDU Artifacts Templates Project Management Section.

Regards,

Unified Process for Education Team

Table of Contents

Software Development Plan

1. Introduction

1.1 Purpose

The objective of this Software Development Plan is to define the activities in terms of the phases and iterations required for developing a Time Monitoring Tool.

1.2 Scope

This Software Development Plan describes the overall plan to be used by the team for developing the Time Monitoring Tool. The details of the individual iterations will be described in the Iteration Plans.
The plans as outlined in this document are based upon the software requirements as defined in the SRS Document.

1.3 Definitions, Acronyms, and Abbreviations

See the Project Glossary

1.4 References

For the Software Development Plan, the list of referenced artifacts includes:

- **SRS**
- **Iteration Plans**
- **Glossary**
- **Configuration Management Plan**
- **Measurement Plan**
- **Project Measurements**
- **Risk List**
- **Test Plan**

1.5 Overview

This *Software Development Plan* contains the following information:

Project Overview	**—**	**provides a description of the project's purpose, scope, and objectives. It also defines the deliverables that the project is expected to deliver.**
Project Organization	**—**	**describes the organizational structure of the project team.**
Management Process	**—**	**explains the estimated schedule, defines the major phases and milestones for the project, and describes how the project will be monitored.**
Technical Process Plans	**—**	**provides an overview of the software development process, including methods, tools and techniques to be followed.**
Supporting Process Plans	**—**	**this includes the configuration management plan.**

2. Project Overview

2.1 Project Purpose, Scope, and Objectives

The Time Monitoring Tool allows developers working within a defined software development process to record the time spent on the various activities, in a database. The TMT will also allow a manager to derive analyses and produce reports based on the data entered in the system.

The Developer Client Module allows developers to log onto the TMT system and to record timestamps corresponding to their activities in a convenient way.

The Manager Client Module allows a manager to retrieve timestamp information from the database to produce analyses and reports. The Developer and Manager Client Modules must provide a user interface that is available through a WWW browser.

2.2 Assumptions and Constraints

The TMT project should be implemented within a 13-week semester. The system must be available in time for the end of the session.

Moreover, the system must offer high security levels using user authentification mechanisms.

Finally, the system must ensure that all stored data keep its integrity.

2.3 Project Deliverables

Deliverables for each project phase are identified in the Development Case. Deliverables are delivered towards the end of the iteration, as specified in section *4.2.4 Project Schedule*.

Requirements

- Supplementary Specifications
- Use-Case Model
- Use-Case Specifications
- Glossary
- User Interface Prototype Document

Analysis & Design

- Analysis Classes
- Design Classes
- Use-Case Realization
- Design Model

Implementation

- Implementation Model (Model and document)
- Components
- Builds and Executables

Tests

- **Defect**
- **Test plan**
- **Test cases**
- **Test Results**
- **Test Components**
- **Test Evaluation Report**

Configuration & Change Management

- **Configuration Management Plan**
- **Workspaces**
- **Project Referential**
- **Change Requests Reports**

Project Management

- **Software Development Plan**
- **Iteration plans**
- **Work Orders**
- **Review Records**
- **Iteration Plans**

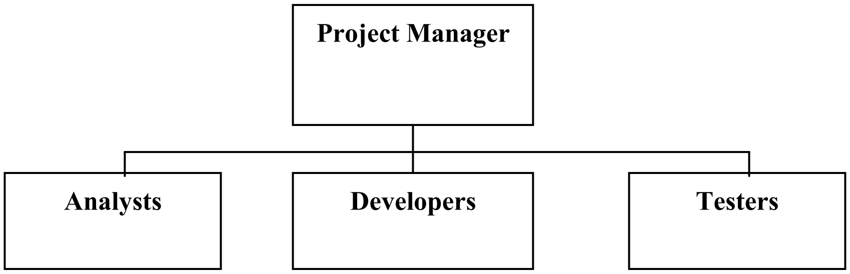
2.4 Evolution of the Software Development Plan

The Software Development Plan will be revised prior to the start of each Iteration phase.

3. Project Organization

3.1 Organizational Structure

The project team for the TMT project will be organized as follows:



3.2 External Interfaces

The project team will interact with the professor and the lab assistant to solicit inputs and review of relevant artifacts.

3.3 Roles and Responsibilities

Role	Responsibility
Project Manager	The Project Manager allocates resources, shapes priorities, coordinates interactions with the customers and users, and generally tries to keep the project team focused on the right goal. The Project Manager also establishes a set of practices that ensure the integrity and quality of project artifacts.
Systems Engineering Team	Primarily responsible for managing the Requirements workflow. Provides support and input to the Project Management workflow.
Software Engineering Team	Primarily responsible for the Analysis and Design, Implementation, Configuration and Change Management, workflows. Provides support and input to the Project Management workflow.
Test Team	Responsible for managing the Test, and Deployment workflows. The Test Team is responsible for test design, test execution, evaluation of test execution and recovery from errors, and assessing the results of test and logging identified defects. Provides support and input to the Project Management workflow.

Anyone on the project can perform Any Role activity.

4. Management Process

4.1 Project Estimates

The time-frame and effort estimates from previous project courses are the basis of this project schedule.

4.2 Project Plan

4.2.1 Phase Plan

The development of the Time Monitoring Tool System will be conducted using a phased approach where multiple iterations occur within a phase. The phases and the relative timeline are shown in the table below:

Phase	No. of Iterations	Start	End
Elaboration Phase	**2**	**Week 1**	**Week 6**
Construction Phase	**2**	**Week 7**	**Week 13**
Transition Phase	**1**	**Week 14**	**Week 15**

The project is divided in 4 iterations with 3 weeks per iteration.

During the first iteration, all needed tools should be gathered and installed; all requirements specifications should be understood and a TMT Client/Server prototype will be developed.

During the second iteration, all major Analysis activities take place. The system Architecture will be defined.

During the third iteration, all major Design activities take place and a critical system will emerge. The more important functionalities will be implemented.

During the fourth iteration, all construction and implementation activities take place and a final system will emerge. All non-functional specifications and less important functionalities will be implemented along with the proper GUI.

Finally, a last iteration will take place for the project's transition, if needed.

Refer to the Iteration Plan Documents for detailed information on each iteration planning.

4.2.2 Iteration Objectives

Each iteration has its own plan de development:

- **TMT_IP1 – first iteration**
- **TMT_IP2 – second iteration**
- **TMT_IP3 – third iteration**
- **TMT_IP4 – fourth iteration**

See the corresponding documents for each iteration

Iteration	Description	Associated Milestone	Addressed Risks
Iteration 1	• Understand the project and analyze the requirements • Realize necessary use-case diagrams and models • Start class diagram designs. • Configuration project repository and workspaces. • Develop the prototype.	Prototype	Clarifies user requirements up front. Develops realistic Software Development Plan. Early prototype for user review
Iteration 2	• Finalize all use-case related artifacts (Artifacts issued from the Analysis activities) • Enhance all project management artifacts • Enhance all configuration management artifacts. • Finalize the Test Plan. • Develope GUI	Architectural Prototype	Architectural issues clarified. Early prototype for user review.
Iteration 3	• Finalize all Design related artifacts • Update all produced artifacts. • Finalize Test Case Artifact. • Implement product functionalities	Prototype	The key features from a user and architectural prospective implemented.

Iteration 4	• Test the product with the elaborated test cases. • Finalize all test-related artifacts. • Finish the product	Final product	Fully reviewed by user Product quality should be high. Defects minimized.

4.2.3 Releases

Prototype: Function Client/Server Interface

Final Release: Complete.

4.2.4 Project Schedule

The high level schedule showing project phases, iterations, and milestones are presented in Section 4.2.2

For more detailed schedules (Each Iteration) Refer to the Iteration Plan Documents.

4.2.5 Project Resourcing

The project team is composed by 5 senior students with appropriate skills.

4.3 Project Monitoring and Control

4.3.1 Requirements Management

The requirements for this system are captured in the SRS document. Supplementary specifications are captured in the Supplementary Specifications document. Requested changes to requirements are captured in Change Requests, and are approved as part of the Configuration Management process.

4.3.2 Quality Control

All deliverables are required to go through the appropriate review process. The review is required to ensure that each deliverable is of acceptable quality, using guidelines and checklists. Formal reviews will be executed for each design and implementation subsystem. This will ensure that the objects under review meet the specified requirements.

4.3.3 *Reporting and Measurement*

Effort and time will be used to track progress of the project. Planned vs. Actual reports will be used by the project manager to measure progress.

Refer to the Measurement Plan Document

4.3.4 Risk Management

Risks will be identified in Inception Phase. Project risk is evaluated at least once per iteration and documented in the Risk List Document.

4.3.5 Configuration Management

Appropriate tools will be selected which provide a controlled versioned repository of project artifacts.

All source code, and data files are included in baselines. Documentation related to the source code is also included in the baseline, such as design documentation. All customer deliverable artifacts are included in the final baseline of the iteration, including executables.

Refer to the Configuration Management Plan Document

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

le génie sans frontières

Time Monitoring Tool
Work Orders (15/01/01 to 21/01/01)

Version <1.0>

Revision History

Date	Version	Description	Author
<12/01/2001>	<1.0>	First Version	John Lemon

Table of Contents

Work Orders (15/01/01 to 21/01/01)

1. Introduction

1.1 Purpose

This document communicates what is to be done, and when, to the responsible staff. It becomes an internal contract between the Project Manager and those assigned responsibility for completion.

1.2 Scope

This document applies to Iteration #1 of the Time Monitoring Tool Project.

1.3 Definitions, Acronyms, and Abbreviations

Refer to the Glossary Document. (See References)

1.4 References

Glossary Document, Glossary, TMT Team 1, École Polytechnique de Montréal, 2001

Iteration Plan Document, Iteration Plan <Iteration #1>, TMT Team 1, École Polytechnique de Montréal, 2001

1.5 Overview

This document first identifies the week of work concerned, the related iterations and the responsibilities implied (Section 2). Then, Work Order Reports are shown in Section 3.

2. Identification

2.1 Week of work

Week #2 - January 2001: 15th to 21st.

2.2 Work Orders ID convention

All work orders ID use the following convention:

<RR-IT#-W##-##**>** where:

RR:	Resource (Work Order Holder) initials
IT#:	**Related Iteration Number**
W##:	**Week of work Number**
##:	**Unique work order number**

2.3 Iteration identification

2.3.1 Iteration name

Iteration #1

2.3.2 Iteration objectives

Main Objective is to develop a TMT Client/Server Prototype

2.4 Responsibilities

2.4.1 Team Members

Each team member has been assigned a role. This role is important in the team's organization but is meaningless when the time to assign the defined tasks comes. Here are the team's responsibilities:

Sandra Lee (SL):	**Documentation manager**
Sabrina Laflamme (SbL):	**Review secretary**
Robert Latour (RL):	**Configuration and environment manager**
James Prévost (JP):	**Timestamps manager**
John Lemon (JL):	**Coordinator** **Project Manager**

NOTE: All team members are skilled enough to perform any task that is not directly linked to his/her responsibility.

2.4.2 Document responsible

The present document is managed by both Environment manager and Project manager: Robert Latour & John Lemon.

3. Work Orders Reports and Schedule

Work Order for Time Monitoring Tool **Created on: <12/01/2001>**

Identification

Work Order ID: <SL-IT1-W02-01>	**WBS ID: <**1.3.8**>**

Responsibility (Holder)

Sandra Lee

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <26/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 10 hours	**Other Resources:** None

Description

Work Description: Develop and implement a functional TMT prototype.

Expected Output(s): TMT Prototype

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <SL-IT1-W02-02>	**WBS ID: <**1.3.9**>**

Responsibility (Holder)

Sandra Lee

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <21/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 3 hours	**Other Resources:** None

Description

Work Description: Plan unit, functional, integration and acceptance tests.

Expected Output(s): Documentation should be included in the Test Plan.

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <SbL-IT1-W02-01>	**WBS ID:** <7.1.1**>**

Responsibility (Holder)

Sabrina Laflamme

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <16/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 2 hours	**Other Resources:** None

Description

Work Description: Define Functional Requirements Classification (Optional, Desirable, Necessary)

Expected Output(s): SRS Document should be updated.

Signature Agreement

Project Manager: x _____________________	**Work Order Holder: x** _____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <SbL-IT1-W02-02>	**WBS ID: <**7.1.4**>**

Responsibility (Holder)

Sabrina Laflamme

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <17/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 2 hours	**Other Resources:** None

Description

Work Description: Define and document Supplementary Specifications

Expected Output(s): Supplementary Specifications Document

Signature Agreement

Project Manager: x _____________________	**Work Order Holder: x** _____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <SbL-IT1-W02-03>	**WBS ID: <**7.1.3**>**

Responsibility (Holder)

Sabrina Laflamme

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <19/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1 hour	**Other Resources:** None

Description

Work Description: Define and Document Glossary Terms (Capture a Common Vocabulary)

Expected Output(s): First version of the Glossary Document.

Signature Agreement

Project Manager: x _	**Work Order Holder: x** _
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <RL-IT1-W02-01>	**WBS ID: <**1.4.1**>**

Responsibility (Holder)

Robert Latour

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <16/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1,5 hours	**Other Resources:** None

Description

Work Description: Write CM Plan

Expected Output(s): First version of the Configuration Management Plan

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <RL-IT1-W02-02>	**WBS ID: <**N/A**>**

Responsibility (Holder)

Robert Latour

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion: <ASAP>**	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1,5 hours	**Other Resources:** None

Description

Work Description: Create Integration and Development Workspaces

Expected Output(s): Accessible Integration Workspaces.

Signature Agreement

Project Manager: x _ **Signed on:** <dd/mm/yy>	**Work Order Holder: x** _ **Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JP-IT1-W02-01>	**WBS ID: <**1.3.2**>**

Responsibility (Holder)

James Prévost

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <17/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1,5 hours	**Other Resources:** None

Description

Work Description: Find actors and Use-Cases

Expected Output(s): Ready to use Actor and use case reports for future Use-Case Specifications Document.

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JP-IT1-W02-02>	**WBS ID: <**N/A**>**

Responsibility (Holder)

James Prévost

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <20/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 5 hours	**Other Resources:** None

Description

Work Description: Create a MicroSoft Access Database for internal timestamps entries.

Expected Output(s): Ready-to-use MicroSoft Access Database for all team members' timestamps entries.

Signature Agreement

Project Manager: x ____________________

Signed on: <dd/mm/yy>

Work Order Holder: x ____________________

Signed on: <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JP-IT1-W02-03>	**WBS ID: <**N/A**>**

Responsibility (Holder)

James Prévost

Associated Change Reports

None

Schedule

Start: <21/01/01>	**Completion:** <21/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1 hour	**Other Resources:** None

Description

Work Description: Include Week #02 team members' timestamps in internal database.

Expected Output(s): Updated Timestamps database.

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JL-IT1-W02-01>	**WBS ID: <**N/A**>**

Responsibility (Holder)

John Lemon

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion: <ASAP>**	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1,5 hours	**Other Resources:** None

Description

Work Description: Get Acquainted with MS-Project Software

Expected Output(s): None

Signature Agreement

Project Manager: x ____________________ **Signed on:** <dd/mm/yy>	**Work Order Holder: x** ____________________ **Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JL-IT1-W02-02>	**WBS ID: <**1.4.2**>**

Responsibility (Holder)

John Lemon

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion: <ASAP>**	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 1,5 hours	**Other Resources:** None

Description

Work Description: Develop Iteration Plan (Iteration #1)

Expected Output(s): Iteration Plan for Iteration #1 and Gantt Chart for Iteration #1

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JL-IT1-W02-03>	**WBS ID: <**1.3.1**>**

Responsibility (Holder)

John Lemon

Associated Change Reports

None

Schedule

Start: <15/01/01>	**Completion:** <18/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 2,5 hours	**Other Resources:** None

Description

Work Description: Model the use cases related to Iteration #1

Expected Output(s): Use-Cases diagrams and Use-Case Model (Packages related to Iteration #1)

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>

Work Order for Time Monitoring Tool

Created on: <12/01/2001>

Identification

Work Order ID: <JL-IT1-W02-04>	**WBS ID: <**1.3.6**>**

Responsibility (Holder)

John Lemon

Associated Change Reports

None

Schedule

Start: <18/01/01>	**Completion:** <21/01/01>	**Critical Path:** <N/A>

Efforts and Other Resources

Staff Hours: 3,5 hours	**Other Resources:** None

Description

Work Description: Analyze and Design the Use-Cases from Iteration #1. Produce appropriate Interaction (Sequence + Collaboration) diagrams.

Expected Output(s): Ready-to-use Interaction Diagrams to be included in the future Use-Case Realizations Document.

Signature Agreement

Project Manager: x ____________________	**Work Order Holder: x** ____________________
Signed on: <dd/mm/yy>	**Signed on:** <dd/mm/yy>