AL AKHAWAYN UNIVERSITY IN IFRANE
SCHOOL OF SCIENCE & ENGINEERING

PROJECT MANAGEMENT SYSTEM: THE CASE OF E-FES PROJECT FROM ENHANCED TECHNOLOGIES

Capstone Design
April 2015
Amine Abbad Andaloussi
Dr. Driss Kettani
ACKNOWLEDGEMENT

I owe debt of gratitude to a number of people without whom this work would not have seen light and whose support and encouragements helped me throughout my stay at Al Akhawayn University.

I would like to express my gratitude to Dr Kettani, my professor, coordinator and supervisor and a role model for me. Thank you for your continuous support and help and all your guidance and assistance during my bachelor degree.

To my little sweet family without whom nothing of this could have been possible. To my dad, thank you for your support, affection and endless love. To my loving and caring mother thank you for being always there for me for your endless love and support. To Imane, my little sweet sister, thank you for being there for me and for your continuous support.

To Alae Gamar who handled me in my worst moments of stress and anger and who held my hand and helped me to overcome obstacles. I am and remain grateful for all your precious help thank you.

To Enhanced Technologies, Mrs Chakiri Houda and Mr Chakiri Zouhair, for their valuable help, kindness, support and commitment. Thank you for being there for me and for helping throughout my project.
## CONTENTS

Acknowledgement ........................................................................................................... i

Contents ........................................................................................................................... ii

List of Figures ................................................................................................................... v

List of Tables ..................................................................................................................... vi

Abstract ........................................................................................................................... vii

1 Introduction .................................................................................................................. 1

2 Feasibility study .......................................................................................................... 2

  2.1 Presenting the Client ............................................................................................... 2

  2.2 Description of the Project ....................................................................................... 2

  2.3 Description de the current system ......................................................................... 3

  2.4 General requirement .............................................................................................. 3

  2.5 The software engineering process ......................................................................... 4

  2.6 Schedule ................................................................................................................. 5

  2.7 Deliverables ............................................................................................................ 5

  2.8 Difficulties ............................................................................................................. 5

3 Steeple analysis .......................................................................................................... 7

4 Description of gathered documents ........................................................................... 8

5 Description of requirements ....................................................................................... 9

  5.1 The Dashboard ..................................................................................................... 9

  5.2 External Data gathering system ........................................................................... 9
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3</td>
<td>Graph generators and simulators</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>Functional requirements</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>Conceptual process models</td>
<td>12</td>
</tr>
<tr>
<td>7.1</td>
<td>Project Management System</td>
<td>12</td>
</tr>
<tr>
<td>7.2</td>
<td>Project Manager Module</td>
<td>13</td>
</tr>
<tr>
<td>7.3</td>
<td>Dashboard Manager</td>
<td>14</td>
</tr>
<tr>
<td>7.4</td>
<td>Graphs and simulator manager</td>
<td>15</td>
</tr>
<tr>
<td>7.5</td>
<td>External Data Gathering System</td>
<td>16</td>
</tr>
<tr>
<td>8</td>
<td>System architecture</td>
<td>18</td>
</tr>
<tr>
<td>8.1</td>
<td>Project management system</td>
<td>18</td>
</tr>
<tr>
<td>8.2</td>
<td>Project management module</td>
<td>19</td>
</tr>
<tr>
<td>8.3</td>
<td>Dashboard management module</td>
<td>19</td>
</tr>
<tr>
<td>8.4</td>
<td>Graphs and simulator module</td>
<td>20</td>
</tr>
<tr>
<td>8.5</td>
<td>External data gathering system module</td>
<td>20</td>
</tr>
<tr>
<td>9</td>
<td>Conceptual data model</td>
<td>21</td>
</tr>
<tr>
<td>9.1</td>
<td>Final version of data dictionary</td>
<td>21</td>
</tr>
<tr>
<td>9.1.1</td>
<td>Data gathered deductively</td>
<td>21</td>
</tr>
<tr>
<td>9.1.2</td>
<td>Data gathered inductively</td>
<td>22</td>
</tr>
<tr>
<td>9.2</td>
<td>Data Entities</td>
<td>23</td>
</tr>
<tr>
<td>9.3</td>
<td>Diagram of conceptual data model</td>
<td>24</td>
</tr>
<tr>
<td>10</td>
<td>Design And Implementation</td>
<td>25</td>
</tr>
</tbody>
</table>
10.1 Algorithms analysis ................................................................. 25
10.2 External data gathering system.................................................. 25
10.3 Database Architecture and Queries .......................................... 31
  10.3.1 Entity relation diagram ....................................................... 31
10.4 SQL stored procedures ............................................................. 32
  10.4.1 Check empty units.............................................................. 32
  10.4.2 Upgrade certificates typed .................................................. 33
  10.4.3 Check limit certificates ...................................................... 33
  10.4.4 Upgrade Unit Progress Percentage ...................................... 34
  10.4.5 Set Daily average ............................................................. 34
  10.4.6 SQL trigger .................................................................. 34
10.5 Web application ................................................................. 35
11 Conclusion ........................................................................... 38
Appendix A Snapshot Excel Requirement Document......................... 39
LIST OF FIGURES

Figure 7.1.1 Conceptual Model for Project Management System .............................. 12
Figure 7.2.1 Conceptual Process Model for Project Manager Module .................... 13
Figure 7.3.1 Conceptual Process Model for Dashboard Manager ............................ 14
Figure 7.4.1 Conceptual Process Model for graphs and simulator manager ............. 15
Figure 7.5.1 Conceptual Process Model for External Data Gathering System ......... 16
Figure 8.1.1: System Architecture for Project management system ....................... 18
Figure 8.2.1: System Architecture for Project Management Module ...................... 19
Figure 8.3.1: System Architecture for Dashboard Management Module ................. 19
Figure 8.4.1 System Architecture for Graphs and Simulator Module .................... 20
Figure 8.5.1 System Architecture for External Data gathering System Module ....... 20
Figure 9.3.1 Diagram of conceptual data model .................................................. 24
Figure 10.2.1 Check Empty Unit Procedure ....................................................... 26
Figure 10.2.2 Upgrade Certificates Typed Procedure ........................................ 27
Figure 10.2.3 Check Limits Certificates Procedure ............................................ 28
Figure 10.2.4 Set Daily Average Procedure ........................................................ 29
Figure 10.2.5 Set Weekly Average Procedure ..................................................... 29
Figure 10.2.6 Upgrade Monthly Average Procedure ........................................... 30
Figure 10.2.7 Upgrade Unit Progress Percentage .............................................. 30
Figure 10.2.8 Upgrade Expected Ending Date .................................................... 31
Figure 10.3.1 Entity Relation Diagram ............................................................... 32
Figure 10.4.1 Snapshot of Check Empty Units Procedure details ......................... 33
Figure 10.4.2 Snapshot of Upgrade Certificates Typed Procedure Details ............ 33
Figure 10.4.3 Snapshot Check Limit Certificates Procedure Details .................. 33
Figure 10.4.4 Snapshot Upgrade Unit Progress Percentage Details .................... 34
Figure 10.4.5  Snapshot Set Daily Average Details................................................................. 34
Figure 10.4.6  Snapshot SQL Trigger Details........................................................................... 35
Figure 10.5.1  Snapshot Verification before adding Unit ......................................................... 36
Figure 10.5.2  Application Resource Properties......................................................................... 37
Figure 10.5.3  Validator-rules.xml........................................................................................... 37
Figure 12.1   Fiche Globale des Projets .................................................................................... 39
Figure 12.2   Fiche Projet........................................................................................................... 39
Figure 12.3   Production Journalière Totale............................................................................... 39

LIST OF TABLES

Table 2.6.1   The schedule of the tasks of the project.............................................................. 5
Table 9.1.1   Table of Data Gathered Deductively ................................................................. 21
Table 9.1.2   Table of Data Gathered Inductively...................................................................... 22
Table 9.2.1   Table of Data Entities ......................................................................................... 23
ABSTRACT

As a step toward the fulfillment of the requirements of the Bachelor of Science in Computer Science comes the following project report. This report summarizes the elaboration of a management system for the implementation of eFez, the Governmental Civil Status from Enhanced Technologies Department. This real time project is an opportunity to acquire a priceless experience and challenge my skills. This project will assist the management of Enhanced Technologies in the implementation of the eFez project and help them to have real time information about the progress of the work done. The methodology used in this project is mainly inspired from the Computer Science courses undertaken during my bachelor degree namely software engineering, Java and the database courses.

This web application contains four modules: the project manager module, the dashboard manager module, the graphs and simulator manager module, and the external data gathering system module. All four modules that were implemented used java as a programming language for the business logic, JavaServer Faces as a component-based user interface, and Microsoft SQL server as database management system (DBMS). The outcome of this project is the project management system web application that will help in managing the implementation of e-Fez.

This project management system will be handled to Enhanced Technologies as a trial version to which the student will bring modifications and adjustments as needed by the users. It will be modified to overcome any last minute problem or modification necessary to the good functioning of the web application.
1 INTRODUCTION

This report presents the work achieved for the accomplishment of my Bachelor of Science in Computer Science. For my capstone project, my supervisor and I chose a challenging and interesting real time project within the Enhanced Technologies Company. This project will allow me to develop my skills and acquire a practical knowledge and priceless experience. In addition, this project is an opportunity to challenge my skills, knowledge and my ability to adapt to real time problems and overcome the time constraint.

The capstone project I will be working on is a project management system for a project of eFez, the Governmental Civil Status from Enhanced Technologies that will assist the management of the company in the implementation and provide them with real time progress of the work accomplished in each Moroccan community.

The project management system is a web application composed of four modules: the project manager module, the dashboard manager module, the graphs and simulator manager module, and the external data gathering system module. These modules were implemented used java as a programming language for the business logic, JavaServer Faces as a component-based user interface, and Microsoft SQL server as database management system (DBMS).

We will start with a feasibility study of the project where we will introduce the Enhanced Technologies, the Governmental Civil Status Project and move to the STEEPLE Analysis, a description of the gathered data, description of requirements, functional requirements, conceptual process model, the system architecture and a conceptual data model in the following chapter of the report.
2 FEASIBILITY STUDY

2.1 Presenting the Client

Enhanced Technologies will host the project I will work on for my capstone. This company was founded in 2005 at Al Akhawayn University and became in less than three years a leader in the Information and Communication Technologies in Morocco. Enhanced Technologies mainly participated in the development of e-solutions for the governmental organisms in Morocco. It counts young engineers, architects, trainers and sociologists that gather multidisciplinary skills and knowledge to enhance the IT environment in Morocco by offering better quality and a better managing system.

2.2 Description of the Project

Enhanced Technologies works on a number of projects from which the e-Fes which is an application that enables the management of the register office for a better management system of the civil status. This application’s purpose is to ease the procedures of registry and delivery of the civil status register, renew the register office services and offer a better data security. The follow up of this project is a difficult task for the management. My capstone project will provide a solution for the management for a continuous and instantaneous follow up of the implementation of e-Fes in multiple communities of Morocco.

This project is a management system for the civil status project of Enhanced Technologies. It will allow the instantaneous follow up of the status of the implementation of the application in different townships of Morocco, including a dashboard that will provide the system user with all data he needs to follow up the project implementation progress and to analyses real time statistics in order to take good managerial decisions.
This project has many important goals among which some personal ones. It will help to sharpen my knowledge and improve my skills in programming. It will provide me with a good opportunity to immerse in the software engineering domain and experience a real time problem and find concrete solutions to overcome the difficulties that I may face during the development of the project.

2.3 Description de the current system

After several meetings with the project manager Mr. Zouhair Chakiri, I could get a global vision of the problems faced in the project. He expressed difficulties to follow the status of implementation of the e-Fes application in the different communities of Morocco. The company uses excel to keep track of all the statistics related to the project, such as the number of birth documents scanned, and proceed for each community. This process is time expensive as it requires a daily update from the employees. However, these data can be retrieved directly from their databases and used in real time. The data gathered will allow an instantaneous follow up of the project, and will be used to provide features such as live graphs and simulators.

2.4 General requirement

The general requirement for the project includes a Dashboard that will comprise detailed and helpful features to the users. This dashboard includes a number of features listed in the following:

- Collect the number of new birth data entries in each township in real-time
- Update statistics based on data gathered
- Provide simulations and statistical graphs to the end user
2.5 The software engineering process

The project has several components that make the specification collection model hard to collect and translate into software that will be useful to the company. To develop an efficient solution I will use the throw-away model that will help to the design and implementation of the solution. This model is flexible to changes and modification however it may cause a problem regarding the specifications. Another model is the waterfall model which is more advanced and complicated model to use for the development of the software solution. None of these two solutions will be used to develop the software. I used the incremental model that will ensure the flexibility of the prototype model and the structure of the waterfall model. The following steps are the main ones used to realize the project:

1. The requirement collection which describes the needs of the client and the aspired tasks from the software.
2. The design that will allow me to follow logic when coding the project. This will allow me to remedy to the problems related to design and the structure of the software.
3. The implementation that needs to respect the requirements and a user friendly interface and a web surface for more visibility.
4. The testing phase will include the testing of the software if it working efficiently and to discover the deficiencies and correct them as needed.
5. The maintenance will allow an update of the website as needed by the client and the correction of any problem whenever it appears.
2.6 Schedule

Table 2.6.1 The schedule of the tasks of the project

<table>
<thead>
<tr>
<th>Week</th>
<th>Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Week 2 (February 2-8 )</td>
<td>Feasibility Study</td>
</tr>
<tr>
<td>Week 3 (February 9 - 15)</td>
<td>Collection of functional requirements</td>
</tr>
<tr>
<td>Week 4 (February 16 - 22)</td>
<td>Elicitation</td>
</tr>
<tr>
<td>Week 5 and 6 (February 23 – march 8)</td>
<td>Achievement of conceptual process models</td>
</tr>
<tr>
<td>Week 8 &amp; 9 (march 9 – march 21)</td>
<td>Achievement of the conceptual data models</td>
</tr>
<tr>
<td>Week 10 (march 22, march 28 )</td>
<td>Design</td>
</tr>
<tr>
<td>Week 11 &amp; Week 12 (march 30, april 5)</td>
<td>Implementation</td>
</tr>
<tr>
<td>Week 13 (April 6, April 12)</td>
<td>testing</td>
</tr>
<tr>
<td>Week 14 (April 13, April 19)</td>
<td>Delivery</td>
</tr>
</tbody>
</table>

The table above shows the scheduled tasks for the capstone project and the length each one will take in order to be completed. I tried to fit within the deadlines in order to make a quality project within the deadlines.

2.7 Deliverables

At the end of the month of April, I will perform the requirement engineering documents and the implementation of the following modules:

- External data gathering System
- Dashboard
- Graph generator and simulator

2.8 Difficulties

The project presented a real challenge concerning the time constraints and the deliverables as well as the implementations and the testing of the software after its implementation.
A number of problems showed during the development process of the software like additions, substitutions or deletions of some modules of the software. The changes of requirements from the client did not affect the delivery time of the software due to the significant care and amount of time I dedicated to work on the project. In addition, the implementation of the project followed the standards set by the company to develop their projects.
3 STEEPLE ANALYSIS

In this chapter we will undertake the STEEPLE analysis. This analysis is an essential component of my project report and to the project elaboration. It will ensure the conformity with the Societal, Technical, Environmental, Ethical, Political, Legal and Economic considerations. Each of the latter components are detailed in the following:

- **Societal:** the project will make the communication of information between the employees easier and help a faster implementation of the e-Fez project that has a greater impact on the society by easy the management of the registry office in Moroccan Provinces.

- **Technical:** This project will help the management to maintain the control over the progress of the e-Fez implementation. This software is also a basis for an ERP system that can be further developed for the entire office and all the projects of the company.

- **Environmental:** The software will lower the dependency on paper use as all the transactions and actions will be done through the software.

- **Ethical:** This software will provide transparency to the management actions and an easier control over the implementation of the project. This software will help the management to respect the deadlines which is one of the core values of Enhanced Technologies.

- **Political:** the software is aligned with the vision of e-Fez which is a project related to the management of the registry office in Moroccan government Provinces.

- **Legal:** the software will use an open source technology enablers that are free and under the general public licenses.

- **Economic:** An increase in the efficiency of implementation of the project in the different provinces will have an economic impact. The faster the implementation the
better because the company will be able to move forward to work on a new implementation.

4 DESCRIPTION OF GATHERED DOCUMENTS

After several meetings with Mr. Zouhair Chakiri, the project manager, I gathered many documents that relate to different aspects of the project. After filtering them, I remain with two documents that are related to my subject.

The first document contains statistics on the number of citizen’s birth data entries entered into the system for each community in a daily and weekly basis. These statistics are used to generate graphs that show the performance of the employees who entered these certificates into the system. The data is gathered from multiple databases. In this context, the software should allow the department to check the statistics in real time and generate graphs to compare the employees’ performance on daily, weekly, and monthly basis for each community.

The second document contains a table with the name of all the communities that participate in the project. Each project is divided into phases. The table provides the achievement status of each phase, and some statistics on them such as the number of certificates agreed, the number of certificates scanned, and the number of certificates entered to the system. The table shows also the starting date, the expected ending date, and the real ending date for each phase. The system should allow the department to add a new project by specifying the community location, and the number of phases. For each phase the department should specify the number of citizen’s birth data entries agreed and the effective starting date then the system will calculate the expected ending date based on the previous statistics.
5 DESCRIPTION OF REQUIREMENTS

5.1 The Dashboard

The dashboard of the project management system gathers statistics of the advancement of the implementation of the project in different communities in Morocco. It allows a timely follow up of the project and an instantaneous addition and modification of the statistics in each step accomplished in the project.

This part of the project includes an operational management side as it allows the computation of the average number of workers, time needed to accomplish tasks as well as the means used to accomplish the task. These different factors will allow an efficient and effective management of the resources used for the implementation of the different parts of the project.

5.2 External Data gathering system

This part refers to the program that will run automatically with a defined frequency of time in order to gathered the required data from different townships databases, and introduce them in the our local database to update statistics.

5.3 Graph generators and simulators

The aim of this module is to assist the end user to deal with the statistics generated. It allows him to generate graphs using the data available in order to follow the variance in the employee’s performance during the implementation of the project. This module can help the managerial department to follow the performance of employees who are assigned to add new birth data entries to the system, and expect date the ending date of each project unit (phase) based on gathered statistics.
6 FUNCTIONAL REQUIREMENTS

The project will be composed of the four following modules:

- Project manager
- Dashboard manager
- Graphs and simulator manager
- External data gathering system

1. Project management

1.1 Add township project information (township name, location, project description…)

1.2 Add township database information

1.2.1 Add community database information (server, username, password, tables prefix …)

1.2.2 Verify connection to external database

1.2.3 Save database information

1.2.3.1 Add new entry to the table

1.3 Add new township unit

1.3.1 Add new township basic information such as unit number, unit starting date, expected unit ending date, number of citizen’s birth data entries agreed, number of citizen’s birth data entries scanned, unit descriptions and notes, township contact name and phone.

1.3.2 Edit township unit basic information

2. Dashboard management

2.1.1 Display dashboard

2.1.1.1 Display projects list

2.1.1.2 Select project
2.1.1.2.1 Display project general information (Name of community, address, location, project description)

2.1.1.2.2 Display unit information (project name, unit number “tranche”, unit status, citizen’s birth data entries agreed “convenus”, citizen’s birth data entries “scanned”, citizen’s birth data entries typed “saisie”, starting entering data “début saisie”, ending data agreed “fin convenue”, project ending date expected “fin projetée”, daily average “moyenne quotidienne”, monthly average “moyenne mensuel”)

3. Graphs and simulator manager

3.1 Generate time - entries average graph
   3.1.1 Select starting and ending period
   3.1.2 Select time scale (days, weeks, months, years)
   3.1.3 Retrieve daily averages from record store
   3.1.4 Compute weekly, monthly, yearly average
   3.1.5 Generate time – entries average chart

3.2 Generate employees - entries average graph
   3.2.1 Select starting and ending period
   3.2.2 Select time scale (days, weeks, months, years)
   3.2.3 Retrieve daily averages for each employee
   3.2.4 Compute weekly, monthly, yearly average
   3.2.5 Order employees by average
   3.2.6 Generate employees – entries average bar chart

4. External data gathering system

4.1 Retrieve database information from local store
4.2 Connect to external database
4.3 Get number of new entries for each employee
4.4 Update records local store with the number of new entries
4.5 Calculate daily entries average based on the number of new entries

7 CONCEPTUAL PROCESS MODELS

7.1 Project Management System

The figure above represents the conceptual model for the whole project management system. It describes the links between processes, users, and stores. The project is composed of 4 modules: The project manager module, the dashboard manager module, the graph and simulator manager module, and the external data gathering system module.

The program will grant access to one type of users named “The management team” They will have access to the four processes previously stated. The first process “Project manager”
will be linked to the project store and units store from which it will add and retrieve all information about projects and corresponding units. The second process “Dashboard manager” will be linked to project store and units store to retrieve data about projects and their corresponding units in order to represent them to the end user. The Third process “Graphs and simulator manager” will be linked to “units average store”, and “unit store” to retrieve daily averages used to generate the needed graphs. The fourth process is an “External data gathering system” linked to ”records store” in order to keep track of the daily number of new entries for each employees. This data is used to populate the “unit average” store with new daily averages.

7.2 Project Manager Module

Figure 7.2.1 Conceptual Process Model for Project Manager Module

Figure 7.2.1 illustrate the conceptual process model for project manager module. It describes the links between end-users, processes and stores. All the three processes are linked to a single end-user (“the management team”). The first process “Add new township project information” allows the end user to add information about the township project such as
township name, township location and project description. It is linked to the project store where all this data is stored. The second process is “Add new township unit”. It is linked to the “Units store”. This process serves to add information related to each unit such as unit number, unit starting date, expected unit ending date, number of citizen’s birth data entries agreed, number of citizen’s birth data entries scanned, unit descriptions and notes, township contact name and phone. The third process is “Add township database information”. It is linked to the “township database connection information store”. This process include the 3 following sub processes: Add township database connection information (server name, username, and password), verify connection to the database, and store them in the “township database connection store”

7.3 Dashboard Manager

Figure 7.3.1 Conceptual Process Model for Dashboard Manager

Figure 7.3.1 illustrate the conceptual process model for the dashboard manager module (process). It includes two processes both of them are linked to the same end user
“Management team”. The first process “Display project general information” is linked to project store. It serves to retrieve project data (previously described) from the project store. The second process “Display unit information” is linked to the units’ store. It serves retrieve unit data (previously described) from the unit store and display them to the end user.

7.4 Graphs and simulator manager

![Conceptual Process Model for graphs and simulator manager](image)

Figure 7.4.1 Conceptual Process Model for graphs and simulator manager

Figure 7.4.1 illustrate the conceptual process model for graphs and simulator manager. Both processes are linked to one end-user “Management team”; and two data stores “Units store”, and “Units average store”.

Below are the common sub processes:
- Select starting and ending period: The end user selects a time interval on which the graph will be generated
- Select time scale: The user selects the time scale of the graph, it can be days, weeks, months, or years
- Retrieve daily averages: The system retrieves the averages from the unit average store according the time interval, and time scale selected

### 7.5 External Data Gathering System

![External Data Gathering System Diagram]

**Figure 7.5.1 Conceptual Process Model for External Data Gathering System**

Figure 7.5.1 illustrate the conceptual process model for external data gathering system. It is called a system, because it works as an independent system that retrieves data from external databases. The external data gathering module can be instantiated either by an end-user call or automatically in a determined frequency of time. The process is linked to the “records store” to update the number of new entries, and the “unit store” to update the unit progress and the unit status. The external data gathering module contain the following sub processes: retrieve
data base information from local store, connect to the external database, get the number of new entries for each employee, update records local store with the number of new entries, calculate daily averages based on the number of new entries, upgrade unit status, and upgrade unit progress.
8 SYSTEM ARCHITECTURE

8.1 Project management system

Figure 8.1.1: System Architecture for Project management system
8.2 Project management module

![Diagram of Project Management Module]

**Figure 8.2.1: System Architecture for Project Management Module**

8.3 Dashboard management module

![Diagram of Dashboard Management Module]

**Figure 8.3.1: System Architecture for Dashboard Management Module**
8.4 Graphs and simulator module

![System Architecture for Graphs and Simulator Module](image)

8.5 External data gathering system module

![System Architecture for External Data gathering System Module](image)
9 CONCEPTUAL DATA MODEL

9.1 Final version of data dictionary

9.1.1 Data gathered deductively

Table 9.1.1 Table of Data Gathered Deductively

<table>
<thead>
<tr>
<th>The attribute</th>
<th>The form</th>
<th>Description</th>
<th>Induction/deduction</th>
<th>C / NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Community name</td>
<td>Fiche global</td>
<td>Community name</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Project unit</td>
<td>Fiche global</td>
<td>A project may be splitted into many units</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>unit status</td>
<td>Fiche global</td>
<td>Status of the unit, pending or achieved</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Certificates agreed</td>
<td>Fiche global</td>
<td>Number of certificates agreed</td>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>Certificates scanned</td>
<td>Fiche global</td>
<td>Number of certificates scanned</td>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>Certificates typed</td>
<td>Fiche global</td>
<td>Number of certificates typed</td>
<td>D</td>
<td>C</td>
</tr>
<tr>
<td>Project typing starting date</td>
<td>Fiche global</td>
<td>Typing starting date which represents the starting date of the project</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Project ending date agreed</td>
<td>Fiche global</td>
<td>The agreed ending date of the project</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Project Ending date expected</td>
<td>Fiche global</td>
<td>The expected ending date of the project</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Monthly average certificates typed</td>
<td>Fiche global</td>
<td>Monthly average certificates typed for each community</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Daily Average certificates typed</td>
<td>Fiche global</td>
<td>Daily Average certificates typed for each community</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Daily total average certificates typed</td>
<td>Production journalière</td>
<td>Daily total average certificates typed</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Weekly total average certificates typed</td>
<td>Production journalière</td>
<td>Weekly total average certificates typed</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Monthly total average certificates typed</td>
<td>Production journalière</td>
<td>Monthly total average certificates typed</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Project description</td>
<td>Fiche projects (G. Info)</td>
<td>Project description</td>
<td>D</td>
<td>NC</td>
</tr>
<tr>
<td>Project client name</td>
<td>Fiche projects (G. Info)</td>
<td>The name of person representing the community</td>
<td>D</td>
<td>NC</td>
</tr>
</tbody>
</table>
### 9.1.2 Data gathered inductively

#### Table 9.1.2 Table of Data Gathered Inductively

<table>
<thead>
<tr>
<th>The attribute</th>
<th>Category</th>
<th>Description</th>
<th>Induction/deduction</th>
<th>C / NC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Community location</td>
<td>Project</td>
<td>The address of the community</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Community database server</td>
<td>Project database</td>
<td>Community database server</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Community database username</td>
<td>Project database</td>
<td>Community database username</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Community database password</td>
<td>Project database</td>
<td>Community database password</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Assignment date</td>
<td>Assign manager to project</td>
<td>Project manager Assignment date</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Assignment status</td>
<td>Assign manager to project</td>
<td>Project manager assignment status</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Assignment ending date</td>
<td>Assign manager to project</td>
<td>Project manager assignment ending date</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Assignment project position</td>
<td>Assign manager to project</td>
<td>Project manager assignment position (chief, client manager, operators manager)</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Number of new entries</td>
<td>Backup records</td>
<td>Daily number of new entries in each external database</td>
<td>I</td>
<td>C</td>
</tr>
<tr>
<td>Update date</td>
<td>Backup records</td>
<td>Date of update of number of entries</td>
<td>I</td>
<td>NC</td>
</tr>
<tr>
<td>Database project Number</td>
<td>Project database</td>
<td>The id of the project in the external database</td>
<td>I</td>
<td>C</td>
</tr>
<tr>
<td>Database unit Number</td>
<td>Unit</td>
<td>The id of unit in the external database</td>
<td>I</td>
<td>C</td>
</tr>
<tr>
<td>Manager First name</td>
<td>Project management team</td>
<td>Manager first name</td>
<td>I</td>
<td>NC</td>
</tr>
</tbody>
</table>
After the construction of the data dictionary, I was obliged to apply some changes to these data in order to come up with efficient data entities. The following section presents the data entities containing the attributes and the entities names as well as the diagram of the conceptual data model.

9.2 Data Entities

Table 9.2.1 Table of Data Entities

<table>
<thead>
<tr>
<th>Entity name</th>
<th>Attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project info</td>
<td>township name, township location, project client name, project chief name, Project description</td>
</tr>
<tr>
<td>Project database</td>
<td>Database server address, Database username, Database password</td>
</tr>
<tr>
<td>Unit</td>
<td>agreed unit starting date, real unit starting date, real unit ending date, expected unit ending date, unit status, unit progress percentage, certificates agreed, certificates scanned, certificates typed, unit description, unit township contact name</td>
</tr>
</tbody>
</table>
### 9.3 Diagram of conceptual data model

![Diagram of conceptual data model](image)

**Figure 9.3.1** Diagram of conceptual data model
10 DESIGN AND IMPLEMENTATION

10.1 Algorithms analysis

Before the implementation of any project, it is important to look at the algorithms that need to be sketched in order to have a clear idea on what the system is supposed to do. In my context, an important part of the system features was obvious and did not require to be detailed in algorithms such as the modules that are responsible for adding, editing, and displaying the projects and their corresponding unit data. Other features such as form verifications and session management were handled by struts framework that provide many pre-built features that need only to be configured using some extensible markup language files. However the external data gathering system module raises a need for algorithms analysis.

10.2 External data gathering system

The external data gathering system is a separated module that works in a determined frequency of time. Its role is to gather new data from external databases, and then upgrade the local database with the new entries.

Theoretically, to ensure software efficiency which one of the fundamental requirement of the project, new data in external databases should be gathered and inserted in the local database in real time, however, in practice this cannot be done, because the process of looking for new data, gathering it, and inserting it in the local database is heavy and time expensive, therefore calling it on each end-user request may overload the server.

In order to gather the data needed for the dashboard efficiently, I decided to create a batch file that will update automatically local database with new data each 24 hours. This solution won’t have bad effect on the database server since the update will be done late at
night or early in the morning where traffic is low. This batch file will contain SQL queries that return the number of new certificate typed by each operator (employee) in the last 24 hours for each township. This number will be entered into a table called “records” of the local database.

This process requires the call of the following procedures:

- **Check_empty_unit():** this procedure serves to set the real unit starting date in project_unit table. First it checks the value of the field “certificate typed” if it is 0, then the current date is set as “real unit starting date”

![Flowchart](image)

**Figure 10.2.1** Check Empty Unit Procedure
- **Upgrade_certificates_typed():** this procedure serves to upgrade the number of certificates typed by adding the number of new entries to the existing ones, and store them in the field “certificates_typed” in table project_unit.

```
get n_new_entries from back_up_records

certificates_typed = certificate Typed + n_new_entries
```

**Figure 10.2.2** Upgrade Certificates Typed Procedure

- **Check_limit_certificates():** this function checks if the number of “certificates typed” reaches the number of “certificates agreed”, if it is the case the “unit status” field change to complete, and the “real_ending_date” field is upgraded with the current date.
Set daily_average() : this function will set the daily average by dividing the number of certificates typed over the number of days since the beginning of the unit.
Figure 10.2.4 Set Daily Average Procedure

- `Set_weekly_average()`: this function will set the weekly average by summing the daily averages of the week and dividing it by 5.

Figure 10.2.5 Set Weekly Average Procedure
- **Upgrade_monthly_average()**: this function will calculate the monthly average by summing the daily averages of 30 days and dividing it by 30.

  ![Diagram](image)

  **Figure 10.2.6 Upgrade Monthly Average Procedure**

- **Upgrade_unit_progress_percentage()**: this function will serve to upgrade the progress percentage of a unit by dividing the number of certificate typed over the number of certificates agreed and multiplying by 100.

  ![Diagram](image)

  **Figure 10.2.7 Upgrade Unit Progress Percentage**
- Upgrade\_expected\_ending\_date(): this function will serve to upgrade the expected ending date by dividing the difference of certificates agreed and certificates typed over the daily average to get an estimation about the number of remaining days before ending the project.

![Diagram showing the process of upgrading expected ending date](image)

**Figure 10.2.8 Upgrade Expected Ending Date**

### 10.3 Database Architecture and Queries

#### 10.3.1 Entity relation diagram

As a first step in the database architecture, I constructed the entity relation diagram (ERD) based on the conceptual data model diagram (CDM) that I realized in the requirement engineering part. The transition from the conceptual data model to the entity relation diagram was a time consuming task. Although they are quietly similar, new tables, columns, and relationships appeared. After a long discussion with my supervisor I get to the following entity relation diagram:
The batch file used to gather data from external databases need to perform several operations on tables in the local databases each time a new entry is found. For this purpose, I implemented the following SQL stored procedures that apply to the algorithms previously stated.

10.4.1 Check empty units

This SQL stored procedure retrieve the number of new certificates typed in the table project unit before updating the number of certificates typed, if the number of certificate
typed is 0, then it set the unit starting date to the current date. The snapshot bellow shows
the procedure details:

```sql
CREATE PROC check_empty_units @unit_id int AS
DECLARE @certificates.Typed int
SET @certificatesTyped = (SELECT certificates Typed FROM project_unit WHERE unit_id = @unit_id)
PRINT @certificatesTyped
IF @certificatesTyped = 0
BEGIN
DECLARE @unit_starting_date date
SET @unit_starting_date = (GETDATE())
UPDATE project_unit SET real_unit_starting_date = @unit_starting_date WHERE unit_id = @unit_id
END
```

**Figure 10.4.1 Snapshot of Check Empty Units Procedure details**

**10.4.2 Upgrade certificates typed**

This procedure upgrade the number of certificates typed, by adding the count of old
entries to the count of new entries. The snapshot bellow shows the procedure details:

```sql
CREATE PROC upgrade_certificates Typed @new_entries int, @unit_id int AS
DECLARE @old_entries int
SET @old_entries = (SELECT certificates Typed FROM project_unit WHERE unit_id = @unit_id)
UPDATE project_unit SET certificates Typed = @old_entries + @new_entries WHERE unit_id = @unit_id
```

**Figure 10.4.2 Snapshot of Upgrade Certificates Typed Procedure Details**

**10.4.3 Check limit certificates**

This procedure check if the count of certificates typed is the same as the count
certificates agreed, if it is the case. It upgrades the real unit ending date and set the unit
status to “complete”. The snapshot bellow shows the procedure details:

```sql
CREATE PROC check_limit_certificates @unit_id int AS
DECLARE @certificates_agreed int
DECLARE @certificatesTyped int
DECLARE @real_unit_ending_date date

SET @certificates_agreed = (SELECT certificates_agreed FROM project_unit WHERE unit_id = @unit_id)
SET @certificatesTyped = (SELECT certificates Typed FROM project_unit WHERE unit_id = @unit_id)
SET @real_unit_ending_date = (SELECT real_unit_ending_date FROM project_unit WHERE unit_id = @unit_id)

PRINT @certificates_agreed
PRINT @certificatesTyped
PRINT @real_unit_ending_date
IF @certificatesTyped >= @certificates_agreed AND @real_unit_ending_date IS NULL
BEGIN
UPDATE project_unit SET real_unit_ending_date = (GETDATE()) WHERE unit_id = @unit_id
UPDATE project_unit SET unit_status = 'Complete' WHERE unit_id = @unit_id
END
```

**Figure 10.4.3 Snapshot Check Limit Certificates Procedure Details**

33
10.4.4 Upgrade Unit Progress Percentage

This SQL stored procedure upgrade the unit progress percentage by dividing the count of certificates typed over the count certificates agreed. The snapshot bellow shows the procedure details:

```
CREATE PROC upgrade_unit_progressPercentage @unit_id int AS

DECLARE @certificates_agreed decimal(18,5)
DECLARE @certificates_typed decimal(18,5)
DECLARE @unit_progress decimal(18,5)

SET @certificates_agreed = (SELECT certificates_agreed FROM project_unit WHERE unit_id = @unit_id)
SET @certificates_typed = (SELECT certificates_typed FROM project_unit WHERE unit_id = @unit_id)
SET @unit_progress = CAST(@certificates_typed/@certificates_agreed AS decimal(18,5))

PRINT @certificates_agreed
PRINT @certificates_typed
PRINT @unit_progress

UPDATE project_unit SET unit_progress = @unit_progress WHERE unit_id = @unit_id
```

**Figure 10.4.4 Snapshot Upgrade Unit Progress Percentage Details**

10.4.5 Set Daily average

This SQL stored procedure allows to set the daily average by dividing the count of certificates typed over the time interval between the unit starting date and the current date. The snapshot bellow shows the procedure details:

```
CREATE PROC set_daily_average @unit_id int AS

DECLARE @certificates_typed decimal(18,5)
DECLARE @real_unit_starting_date date
DECLARE @period decimal(18,5)

SET @certificates_typed = (SELECT certificates_typed FROM project_unit WHERE unit_id = @unit_id)
SET @real_unit_starting_date = (SELECT real_unit_starting_date FROM project_unit WHERE unit_id = @unit_id)
SET @period = DATEDIFF(DAY, @real_unit_starting_date, GETDATE()) + 1

INSERT INTO Unit_Averages (unit_id, date, daily_average) VALUES (@unit_id, GETDATE(), @certificates_typed / @period)
```

**Figure 10.4.5 Snapshot Set Daily Average Details**

10.4.6 SQL trigger

In order to run the SQL stored procedures, I create an SQL trigger that runs after each insertion of the “backup records” table. The snapshot bellow shows the trigger in details
Nowadays, Most of business solutions are built on the web platform because this last ensure two important software features portability and usability. The web application consists of developing two sides, the server side which contains the business logic and the client side that contains the user interface. In the client side the web browser load the user interface. This architecture facilitates the user interaction, and make the application accessible on any machine that supports web browsers. In addition, web applications offer other advantages over desktop applications. In term of maintenance, the web application is installed once on the server side, hence, changes and updates in the applications does not affect the end user, Moreover, in term of portability the end user can access the web applications from any location and on any machine which allows multiple end users to work simultaneously on the same application.

**Figure 10.4.6   Snapshot SQL Trigger Details**

10.5 Web application

```sql
CREATE TRIGGER newentries ON backup_records FOR INSERT AS
DECLARE @unit_id INT
DECLARE @n_new_entries INT
DECLARE @unit_number INT
DECLARE @project_id INT
DECLARE @project_number INT
SET @project_number = (SELECT project_number FROM INSERTED)
SET @project_id = (SELECT p_info_id FROM project_info WHERE db_project_number = @project_number)
SET @unit_number = (SELECT unit_number FROM INSERTED)
SET @n_new_entries = (SELECT n_new_entries FROM INSERTED)
SET @unit_id = (SELECT unit_id FROM project_unit WHERE unit_number = @unit_number AND project_id = @project_id)
print @project_number
print @project_id
print @unit_number
print @unit_id
EXEC check_empty_units @unit_id
EXEC upgrade_certificates_typed @n_new_entries, @unit_id
EXEC check_limit_certificates @unit_id
EXEC upgrade_unit_progress_percentage @unit_id
EXEC set_daily_average @unit_id
```
The project management system is implemented on a web platform using Java to implement the business logic, and JavaServer Pages (JSP) to manage the interactions with the end user. To optimize the user interaction with the system I choose STRUTS as an MVC framework that offers many pre-built features that can be configured using extensible markup language files.

One of the most powerful features in STRUTS is the form verification which can be done simply by linking the end user form with a validator that checks if the input fields respect a set of predefined rules. As an example of the struts form verification, the following snapshot shows the form verification done before adding a new unit.

```
<form name="NewUnitForm">
  <field
    property="num"
    depends="required, integer">
    <arg key="label.num"/>
  </field>
  <field
    property="startdate"
    depends="required, date">
    <arg key="label.startdate"/>
  </var>
  <var-name>dataPatterns</var-name>
  <var-value>yyyy-MM-dd</var-value>
</var>
</field>
  <field
    property="enddate"
    depends="date">
    <arg key="label.enddate"/>
  </var>
  <var-name>datePattern</var-name>
  <var-value>yyyy-MM-dd</var-value>
</var>

  <field
    property="nagreed"
    depends="required, integer">
    <arg key="label.nagreed"/>
  </field>
  <field
    property="unagreed"
    depends="required, integer">
    <arg key="label.unagreed"/>
  </field>
  <field
    property="urphone"
    depends="mask">
    <arg key="label.urphone"/>
  </var>
  <var-name>mask</var-name>
  <var-value>[0-9]*</var-value>
</var>
</form>
```

**Figure 10.5.1  Snapshot Verification before adding Unit**
In the snapshot above, for each input field “<field>“, the attribute “property” contains the name of input field in the form, the attribute “depends” contains the rules to be checked, and the attribute “key” contains a pointer to the field name to be displayed to the end user in case the verification fails. The following snapshot shows some error messages that are displayed to the user in case the verification fails.

```
# -- validator --
errors.invalid[0] is invalid.
errors.maxlength[0] can not be greater than {1} characters.
errors.minlength[0] can not be less than {1} characters.
errors.range[0] is not in the range {1} through {2}.
errors.required[0] is required.
errors.byte[0] must be an byte.
errors.date[0] is not a date.
errors.double[0] must be an double.
```

**Figure 10.5.2 Application Resource Properties**

And finally, the following snapshot shows how struts call pre-built classes to operate checking.

```xml
<validator name="required"
    className="org.apache.struts.validator.FieldChecks"
    method="validateRequired"
    methodParam="java.lang.Object,
    org.apache.commons.validator.ValidatorAction,
    org.apache.commons.validator.Field,
    org.apache.struts.action.ActionMessages, 
    org.apache.commons.validator.Validator,
    javax.servlet.http.HttpServletRequest"
    msg="errors.required"/>

<validator name="mask"
    className="org.apache.struts.validator.FieldChecks"
    method="validateMask"
    methodParam="java.lang.Object,
    org.apache.commons.validator.ValidatorAction,
    org.apache.commons.validator.Field,
    org.apache.struts.action.ActionMessages,
    org.apache.commons.validator.Validator,
    javax.servlet.http.HttpServletRequest"
    msg="errors.invalid"></validator>

<validator name="date"
    className="org.apache.struts.validator.FieldChecks"
    method="validateDate"
    methodParam="java.lang.Object,
    org.apache.commons.validator.ValidatorAction,
    org.apache.commons.validator.Field,
    org.apache.struts.action.ActionMessages,
    org.apache.commons.validator.Validator,
    javax.servlet.http.HttpServletRequest"
    msg="errors.date">
    <functionName="DateValidations"/>
```

**Figure 10.5.3 Validator-rules.xml**
11 CONCLUSION

The project management system is a web application that will enable the management team of enhanced technologies to follow up the implementation progress of their projects in different townships in a real time. It is an important plus for their company as it will assist them during the implementation process and allow them to monitor different parameters. This web application contains four modules: the project manager module, the dashboard manager module, the graphs and simulator manager module, and the external data gathering system module. All the four modules were implemented using java as a programming language for the business logic, JavaServer Faces as a component-based user interface, and Microsoft SQL server as database management system (DBMS). However, this application is under the testing phase and will be delivered to enhanced technologies before the capstone presentation to get sufficient time for testing and modifying the different features in order to deliver a final product that will suit the needs of the different users within the company.
APPENDIX A SNAPSHOT EXCEL REQUIREMENT DOCUMENT

Figure 12.1  Fiche Globale des Projets

Figure 12.2  Fiche Projet

Figure 12.3  Production Journalière Totale